

TRABALHO DE GRADUAÇÃO

Open FluidSim: uma ferramenta multiplataforma para sistemas hidráulicos e pneumáticos

Gabriel Naves da Silva

Brasília, Julho de 2018



**ENGENHARIA
MECATRÔNICA**
UNIVERSIDADE DE BRASÍLIA

UNIVERSIDADE DE BRASÍLIA
Faculdade de Tecnologia
Curso de Graduação em Engenharia de Controle e Automação

TRABALHO DE GRADUAÇÃO

***Open FluidSim*: uma ferramenta
multiplataforma para sistemas
hidráulicos e pneumáticos**

Gabriel Naves da Silva

*Relatório submetido como requisito parcial de obtenção
de grau de Engenheiro de Controle e Automação*

Banca Examinadora

Profa. Aida Alves Fadel, ENM/UnB
Orientadora

Profa. Carla Denise Castanho, CIC/UnB
Examinador interno

Prof. Alexandre Zaghetto, CIC/UnB
Examinador interno

Brasília, Julho de 2018

FICHA CATALOGRÁFICA

GABRIEL, NAVES DA SILVA

Open FluidSim: uma ferramenta multiplataforma para sistemas hidráulicos e pneumáticos

[Distrito Federal] 2018.

vii, 55p., 297 mm (FT/UnB, Engenheiro, Controle e Automação, 2018). Trabalho de Graduação – Universidade de Brasília. Faculdade de Tecnologia.

1. Hidráulica

2. Pneumática

3. Simulação

4. Modelagem

I. Mecatrônica/FT/UnB

REFERÊNCIA BIBLIOGRÁFICA

NAVES, G. (2018). Open FluidSim: uma ferramenta multiplataforma para sistemas hidráulicos e pneumáticos. Trabalho de Graduação em Engenharia de Controle e Automação, Publicação FT.TG-*n*º9, Faculdade de Tecnologia, Universidade de Brasília, Brasília, DF, 65p.

CESSÃO DE DIREITOS

AUTOR: Gabriel Naves da Silva

TÍTULO DO TRABALHO DE GRADUAÇÃO: Open FluidSim: uma ferramenta multiplataforma para sistemas hidráulicos e pneumáticos

GRAU: Engenheiro

ANO: 2018

É concedida à Universidade de Brasília permissão para reproduzir cópias deste Trabalho de Graduação e para emprestar ou vender tais cópias somente para propósitos acadêmicos e científicos. O autor reserva outros direitos de publicação e nenhuma parte desse Trabalho de Graduação pode ser reproduzida sem autorização por escrito do autor.

Gabriel Naves da Silva

Colônia Agrícola Águas Claras chácara 40, lote 18.

71090-475 Brasília – DF – Brasil.

RESUMO

O estudo de sistemas hidráulicos e pneumáticos é requisito para os alunos de Engenharia Mecatrônica da Universidade de Brasília. Entretanto, não existe atualmente nenhuma ferramenta de edição e simulação de circuitos hidráulicos e pneumáticos que seja livre e multiplataforma. Existem soluções educacionais proprietárias, mas para os indivíduos não vinculados a instituições de ensino com acesso a tais ferramentas não há nenhuma alternativa. Esse trabalho propõe o desenvolvimento da primeira ferramenta livre e multiplataforma de simulação de sistemas hidráulicos e pneumáticos, o *Open FluidSim*. Foi criada uma interface gráfica unificada para todas as plataformas, e a verificação do funcionamento dos algoritmos de simulação criados foi feita por meio de análise do comportamento em simulação dos elementos fluidos em um conjunto de circuitos de teste. Foi possível modelar todos os circuitos de teste no *software* desenvolvido. No que diz respeito ao comportamento em simulação, simplificações nos algoritmos de simulação dos elementos fluidos resultaram em um comportamento diferente do previsto para um dos seis circuitos de teste.

Palavras Chave: sistemas hidráulicos, pneumáticos, simulador, modelagem, código aberto

ABSTRACT

The study of hydraulic and pneumatic systems is a requisite for all Mechatronics Engineering students at Universidade de Brasília. However, there are no open-source and multiplatform tools for the modeling and simulation of hydraulic and pneumatic circuits. Proprietary educational solutions do exist, but for individuals without connections to learning institutes with access to these tools there are no alternatives. This work proposes the development of the first open-source, multiplatform, hydraulics and pneumatics simulator, named *Open FluidSim*. A unified graphical user interface was designed for all platforms, and the created simulation algorithms were tested by means of analysis of the simulated behaviour of the fluid elements in a set of test circuits. All circuits were correctly modeled using the developed tool. Regarding simulation behaviour, simplifications made on the simulation algorithms for fluid elements resulted in a different simulated behaviour, when compared to the expected one, for one of the six test circuits.

Keywords: hydraulic, pneumatic systems, simulator, modeling, open-source

SUMÁRIO

1	INTRODUÇÃO	1
1.1	SISTEMAS FLUIDO MECÂNICOS	1
1.2	ENSINO DE HIDRÁULICA E PNEUMÁTICA NA UNIVERSIDADE DE BRASÍLIA	2
1.3	DEFINIÇÃO DO PROBLEMA	2
1.4	OBJETIVOS DO PROJETO	3
1.4.1	OBJETIVO GERAL	3
1.4.2	OBJETIVOS ESPECÍFICOS	3
1.5	ESTRUTURA DO TRABALHO	3
2	COMPARAÇÃO DE FERRAMENTAS EXISTENTES	4
2.1	SIMULAÇÃO DE SISTEMAS FLUIDO MECÂNICOS NO <i>Automation Studio</i>	4
3	ESTRUTURA DO <i>software Open FluidSim</i>	8
3.1	DEFINIÇÃO DO ESCOPO DE ATUAÇÃO DO SOFTWARE	8
3.2	ELEMENTOS BÁSICOS DE SISTEMAS HIDRÁULICOS E PNEUMÁTICOS	8
3.2.1	BOMBAS HIDRÁULICAS E RESERVATÓRIOS	9
3.2.2	COMPRESSORES E EXAUSTORES	9
3.2.3	ATUADORES LINEARES	10
3.2.4	VÁLVULAS DIRECIONAIS	11
3.2.5	CONTROLE ELÉTRICO DE SISTEMAS FLUIDO MECÂNICOS	12
3.3	ESPECIFICAÇÃO DA BIBLIOTECA DE ELEMENTOS	13
3.4	REQUISITOS DO SOFTWARE	14
3.5	ESTRUTURA GERAL DO SOFTWARE	16
3.6	APRESENTAÇÃO DA INTERFACE GRÁFICA	18
3.6.1	BIBLIOTECAS DE ELEMENTOS	19
3.6.2	PAINEL DE EDIÇÃO DE CIRCUITOS	19
3.6.3	MODO DE SIMULAÇÃO DE CIRCUITOS	20
4	IMPLEMENTAÇÃO	22
4.1	A <i>engine Unity</i>	22
4.1.1	NOÇÕES BÁSICAS DE <i>Unity</i>	22
4.1.2	<i>Scripting</i> EM <i>Unity</i>	24
4.1.3	<i>JSON Utility</i>	25

4.2	<i>Design</i> DE BAIXO NÍVEL.....	26
4.2.1	MODELAGEM DO ELEMENTOS NO <i>Open FluidSim</i>	26
4.2.2	CONEXÕES ENTRE ELEMENTOS E REPRESENTAÇÃO DE FIOS E TUBULAÇÕES ...	27
4.2.3	FUNCIONAMENTO DA FUNÇÃO DE SALVAR E CARREGAR CIRCUITOS.....	29
4.2.4	FUNCIONAMENTO DO HISTÓRICO DE AÇÕES.....	30
4.3	ESPECIFICAÇÃO DOS ALGORITMOS DE SIMULAÇÃO.....	31
4.3.1	SIMULAÇÃO DOS CIRCUITOS ELÉTRICOS.....	31
4.3.2	SIMULAÇÃO DOS CIRCUITOS FLUIDOS.....	33
5	VALIDAÇÃO DO SIMULADOR - TESTES	35
5.1	METODOLOGIA DE IMPLEMENTAÇÃO DOS TESTES.....	35
5.1.1	DIAGRAMA TRAJETO-PASSO.....	35
5.2	TESTES PROPOSTOS.....	36
5.3	DISCUSSÃO DOS RESULTADOS DOS TESTES.....	38
6	CONCLUSÕES E PROPOSTAS DE TRABALHOS FUTUROS.....	50
6.1	TRABALHOS FUTUROS.....	51
	REFERÊNCIAS BIBLIOGRÁFICAS	52

LISTA DE FIGURAS

2.1	Visão inicial do <i>Automation Studio</i>	5
2.2	Visão das bibliotecas de elementos do <i>Automation Studio</i>	6
2.3	Modelagem dos circuitos no <i>Automation Studio</i>	6
2.4	Simulação do sistema no <i>Automation Studio</i>	7
3.1	Simbologia de bombas e reservatórios	9
3.2	Simbologia de bombas e reservatórios	10
3.3	Esquema simplificado de um cilindro [1]	10
3.4	Cilindro de simples ação com retorno por mola [1]	11
3.5	Diagrama simplificado de exemplo do funcionamento interno de uma válvula direcional [1]	11
3.6	Simbologia de contadores e contatos	12
3.7	Diagrama de um contator [2]	13
3.8	Simbologia de contadores e contatos	13
3.9	Simbologia de sensores	13
3.10	Diagrama do <i>design</i> de alto nível do simulador	17
3.11	Interface gráfica do simulador, visão do modo de editor	18
3.12	Interface das três bibliotecas de elementos, e visão de um elemento flutuante	19
3.13	Visão das diferentes funcionalidades de edição	20
3.14	Interface gráfica do simulador, visão do modo de editor	21
4.1	O editor <i>Unity</i>	23
4.2	Adição de um <i>UI Button</i> na tela	24
4.3	Adição de um <i>script</i> a um objeto	25
4.4	Representação do cilindro pneumático de dupla ação dentro do simulador	27
4.5	Exemplo de conexão entre elementos pneumáticos com tubulação complexa no <i>Automation Studio</i>	28
4.6	Diagramas do funcionamento das conexões entre elementos	28
5.1	Diagrama trajeto-passo de um sistema arbitrário	36
5.2	Diagramas do primeiro sistema de teste	37
5.3	Simulação do primeiro circuito de teste	40
5.4	Diagramas pneumático e elétrico do segundo circuito de teste	41
5.5	Oscilação do contator <i>K1</i> na simulação do teste da Figura 5.4 no <i>Automation Studio</i>	41

5.6	Diagramas pneumático e elétrico do terceiro circuito de teste	41
5.7	Diagramas do quarto sistema de teste.....	42
5.8	Diagramas hidráulico e elétrico do quinto sistema de teste	42
5.9	Diagramas hidráulico e elétrico do sexto sistema de teste.....	43
5.10	Simulação do primeiro circuito de teste.....	44
5.11	Simulação do segundo circuito de teste	45
5.12	Simulação do terceiro circuito de teste	46
5.13	Simulação do quarto circuito de teste	47
5.14	Simulação do quinto circuito de teste	48
5.15	Simulação do sexto circuito de teste	49

LISTA DE TABELAS

2.1	Tabela de ferramentas disponíveis para sistemas fluidos.....	4
3.1	Tabela de elementos da biblioteca hidráulica	14
3.2	Tabela de elementos da biblioteca pneumática.....	15
3.3	Tabela de elementos compartilhados entre as bibliotecas hidráulica e pneumática	15
3.4	Tabela de elementos da biblioteca elétrica	16

LISTA DE SÍMBOLOS

Siglas

UnB	Universidade de Brasília
API	<i>Application Programming Interface</i>
JSON	<i>JavaScript Object Notation</i>
JIC	<i>Joint Industrial Council</i>

Capítulo 1

Introdução

Por séculos a humanidade se utilizou do ar e da água para diversas tarefas. O moinho de vento, construção projetada para converter a energia cinética de correntes de ar em movimento mecânico, encontra seu primeiro uso historicamente aceito no século X, na região da Pérsia [3]. Já o moinho de água, que utiliza a energia da água de córregos e rios para gerar movimento mecânico, teve seu funcionamento básico descrito pelo arquiteto e engenheiro romano Vitrúvio, por volta de 25 AC [4]. E séculos antes, o grego Arquimedes documentou um dispositivo capaz de bombear água para posições mais elevadas, o parafuso de Arquimedes, que ainda encontra aplicações contemporâneas [5].

Em 1795 o inglês *Joseph Bramah* inventou a prensa hidráulica, uma máquina capaz de gerar uma força compressiva por meio de cilindros hidráulicos [6]. Utilizando a água como meio de transmissão de energia, e o princípio de Pascal [1], a máquina amplifica uma força aplicada em um cilindro, resultando em uma força compressiva no outro.

Já no ano de 1868 o americano *George Westinghouse* inventou o freio a ar, um dispositivo mecânico que utiliza ar comprimido para pressionar o pistão de um cilindro pneumático contra a pastilha de freio de um veículo. Originalmente projetado para a frenagem de trens, também é utilizado em caminhões, ônibus e outros veículos de grande porte [7].

1.1 Sistemas fluido mecânicos

De modo geral, sistemas que utilizam fluidos como forma de transmissão de energia com intuito de aplicar uma força, ou produzir movimento, são chamados de sistemas fluido mecânicos. Quando o fluido utilizado é um líquido, como a água ou óleos, o sistema denomina-se hidráulico. Em ocasião do uso de ar comprimido, o sistema denomina-se pneumático [1].

A escolha do fluido utilizado no sistema influencia diretamente suas aplicações. O uso de líquidos incompressíveis em sistemas hidráulicos permite a manipulação de grandes cargas. Além disso, atuadores hidráulicos apresentam uma alta precisão de velocidade e posicionamento, podendo controlar sua posição com uma exatidão expressa em micrômetros [8] [9].

Por outro lado, sistemas pneumáticos, que utilizam gases compressíveis, possuem outras características. Atuadores pneumáticos apresentam uma atuação mais suave quando comparado com os hidráulicos. Eventuais choques no sistema são amortecidos pelo próprio gás utilizado. Em contrapartida, o posicionamento preciso de atuadores pneumáticos é dificultado por essa característica do fluido. São comumente utilizados onde se deseja movimento de duas posições apenas, início e fim [1] [10].

1.2 Ensino de hidráulica e pneumática na Universidade de Brasília

O Departamento de Engenharia Mecânica da Universidade de Brasília fornece a seus alunos o curso de Sistemas Hidráulicos e Pneumáticos [11], que é uma disciplina obrigatória para os alunos de Engenharia Mecatrônica [12] e optativa para os de Engenharia Mecânica [13].

Esta disciplina aborda os principais conceitos de sistemas fluido mecânicos: seu funcionamento básico, elementos utilizados, leitura e interpretação de plantas, e projeto, simulação e montagem de circuitos [11]. O laboratório da disciplina fornece computadores com o *software Automation Studio* instalado para a modelagem e simulação dos circuitos estudados, bem como bancadas de treinamento e diversos elementos para que os alunos possam testar os circuitos estudados em um ambiente didático.

1.3 Definição do problema

Atualmente, uma versão desatualizada do *Automation Studio* é disponibilizada aos alunos do curso de Sistemas Hidráulicos e Pneumáticos da Universidade de Brasília por meio do laboratório de sistemas hidráulicos e pneumáticos da universidade. Entretanto, o laboratório dispõe de uma quantidade limitada de computadores com o simulador disponível, além de uma quantidade limitada de elementos fluido mecânicos e bancadas de montagem.

Consideradas as limitações de recursos disponíveis no laboratório, diversos alunos buscam ter o simulador em seus computadores pessoais, para auxiliar nos estudos quando fora do horário de aula. Existe uma versão educacional do *Automation Studio*, mas esta não é voltada aos alunos, e sim às instituições de ensino. O estudante pode ter acesso remoto ao software instalado no servidor da universidade, mas para tal é necessário que a instituição adquira a versão educacional, que não é gratuita. [14]

Existem outras alternativas para a simulação de sistemas fluido mecânicos. A *MathWorks* possui uma biblioteca para modelagem e simulação destes sistemas, a *Simscape Fluids* [15]. A *MSC Software* possui o *Easy5* [16], que também simula sistemas hidráulicos e pneumáticos, e a *Siemens* possui o *Simcenter Amesim* [17]. Nenhuma destas alternativas é gratuita, e apenas o *Simscape Fluids* pode ser comprado por indivíduos, além de empresas.

Não existe até o momento nenhum *software* livre e multiplataforma para simulação de sistemas fluido mecânicos. Enquanto que alunos de instituições com acesso à versão educacional do *Auto-*

mation Studio possuem uma ferramenta de aprendizado adequada, o restante dos estudantes, bem como indivíduos não vinculados a nenhuma instituição de ensino, podem encontrar um obstáculo financeiro significativo em seus estudos.

1.4 Objetivos do projeto

1.4.1 Objetivo geral

O objetivo do presente trabalho é conceber e implementar o primeiro editor e simulador de sistemas hidráulicos e pneumáticos multiplataforma e de código aberto, o *Open FluidSim*.

O projeto do simulador não visa a competição com *softwares* de simulação previamente estabelecidos no mercado. Ao invés disso, objetiva constituir uma ferramenta livre de aprendizado para qualquer indivíduo que deseje aprender o funcionamento básico de sistemas fluido mecânicos.

1.4.2 Objetivos específicos

1. Estudar e selecionar as ferramentas computacionais disponíveis para a implementação do simulador;
2. Listar os elementos mínimos necessários para a implementação de um modelo funcional de simulação;
3. Projetar e implementar uma interface de usuário simples e intuitiva para o ambiente de simulação, contemplando as principais funcionalidades de edição esperadas de ferramentas do gênero, além dos requisitos específicos das diferentes plataformas alvo;
4. Especificar algoritmos de simulação dos sistemas abordados;
5. Construir os elementos hidráulicos, pneumáticos e elétricos no ambiente virtual;
6. Definir e implementar o código para funcionamento de cada elemento simulável.

1.5 Estrutura do trabalho

Este trabalho divide-se em seis capítulos, incluindo a Introdução. O Capítulo 2 apresenta um estudo comparativo das ferramentas existentes para modelagem e simulação de sistemas fluidos. O Capítulo 3 apresenta o projeto do simulador a ser desenvolvido, o *Open FluidSim*. O Capítulo 4 apresenta as principais decisões de implementação tomadas durante o desenvolvimento do simulador. O Capítulo 5 especifica uma metodologia de validação do correto funcionamento da ferramenta desenvolvida, utilizando circuitos de teste. Finalmente, o Capítulo 6 apresenta as conclusões finais e sugestões de trabalhos futuros.

Capítulo 2

Comparação de ferramentas existentes

Existem diversas ferramentas disponíveis para modelagem e simulação de sistemas hidráulicos e pneumáticos. Neste capítulo efetua-se uma rápida comparação destas ferramentas. Em seguida, discute-se acerca do processo de simulação de sistemas fluidos no *software Automation Studio*.

A Tabela 2.1 fornece uma visão geral de algumas das principais ferramentas disponíveis. Destas, a *QuickDesign* da *Sun Hydraulics* é para navegador, e as outras são para *Windows*. O *Model Selection Software*, da *SMC Corporation* suporta apenas elementos pneumáticos.

Tabela 2.1: Tabela de ferramentas disponíveis para sistemas fluidos

Nome	Fabricante	Modelagem	Simulação	Gratuita
<i>Automation Studio</i> [18]	<i>Famic Technologies Inc.</i>	Sim	Sim	Não
<i>FluidSIM 5</i> [19]	<i>Festo</i>	Sim	Sim	Não
<i>Simscape Fluids</i> [15]	<i>Mathworks</i>	Sim	Sim	Não
<i>Easy5</i> [16]	<i>MSC Software</i>	Sim	Sim	Não
<i>Simcenter Amesim</i> [17]	<i>Siemens</i>	Sim	Sim	Não
<i>Scheme Editor</i> [20]	<i>Bosch Rexroth</i>	Sim	Não	Sim
<i>Model Selection Software</i> [21]	<i>SMC Corporation</i>	Sim	Não	Sim
<i>QuickDesign</i> [22]	<i>Sun Hydraulics LLC</i>	Sim	Não	Sim

2.1 Simulação de sistemas fluido mecânicos no *Automation Studio*

Conforme mencionado na seção 1.2, utiliza-se o *Automation Studio* no laboratório de Sistemas Hidráulicos e Pneumáticos da Universidade de Brasília para realizar as simulações dos sistemas fluido mecânicos estudados. Esta seção apresenta a sequência de passos necessária para realizar uma simulação neste *software*.

A Figura 2.1 mostra a tela de novo projeto do *Automation Studio*. Para iniciar a construção dos circuitos, o usuário deve adicionar uma biblioteca válida de elementos. Esta consiste em um arquivo externo, que deve ser localizado na máquina do usuário e carregado. O botão de abrir a

janela de elementos está evidenciado em vermelho.

A Figura 2.2 mostra a janela da biblioteca de elementos com um arquivo carregado. O botão em vermelho possui a função de carregar novas bibliotecas, e várias destas podem estar carregadas simultaneamente. Para adicionar um elemento ao circuito o usuário deve clicar no ícone correspondente e arrastá-lo para dentro do projeto.

A Figura 2.3 mostra um circuito qualquer modelado no *software*. O botão de início de simulação está marcado em vermelho.

A Figura 2.4 mostra um momento não arbitrário na execução desta simulação. O *Automation Studio* apresenta um atraso considerável na identificação de eventos de sensores. Na imagem, o cilindro da direita se localiza próximo do fim de seu trajeto de retorno, mas o sensor 2 ainda o identifica como se estivesse na outra ponta de seu trajeto.

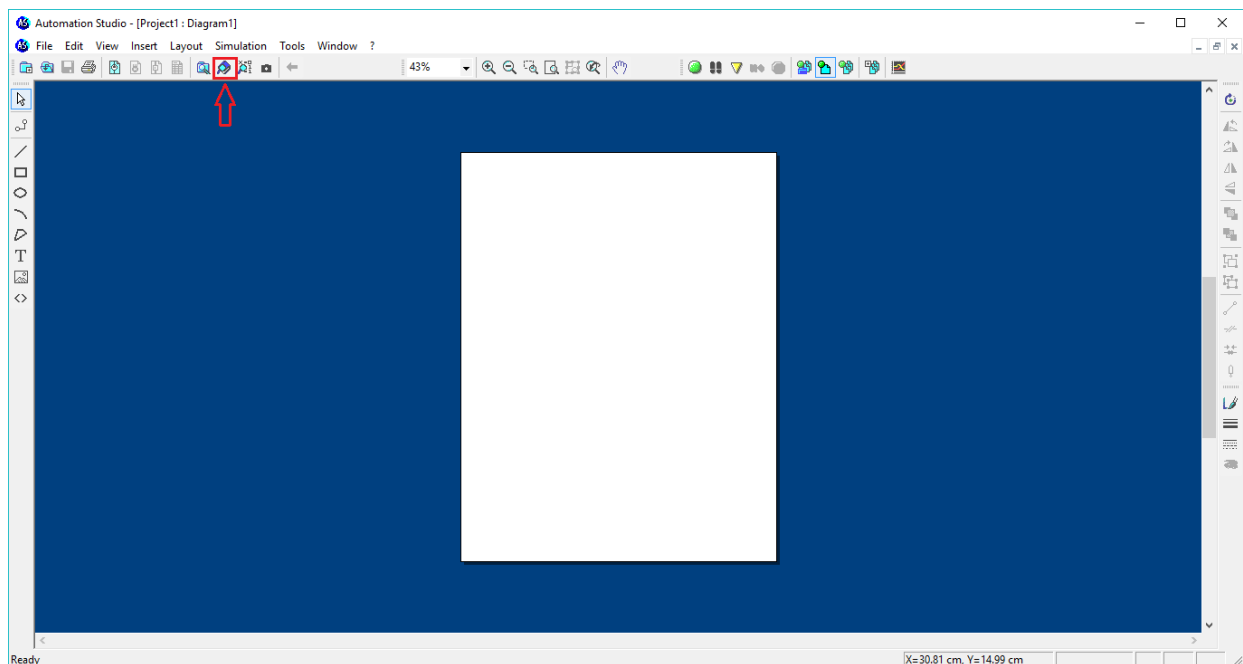


Figura 2.1: Visão inicial do *Automation Studio*

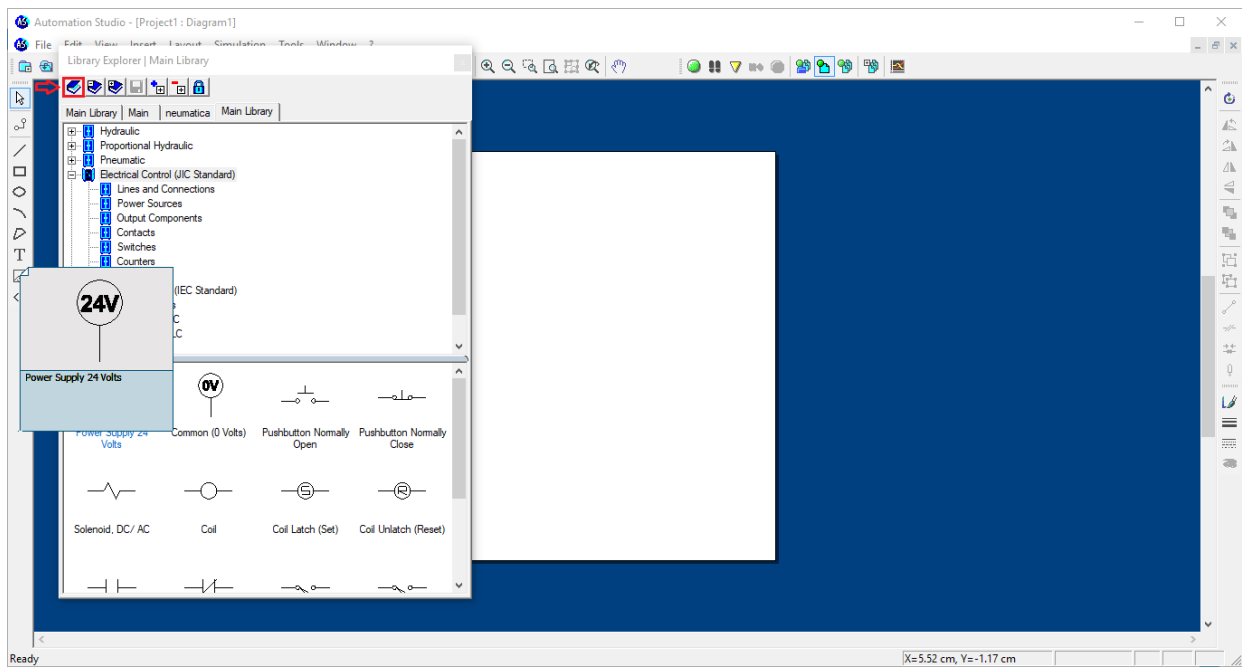


Figura 2.2: Visão das bibliotecas de elementos do *Automation Studio*

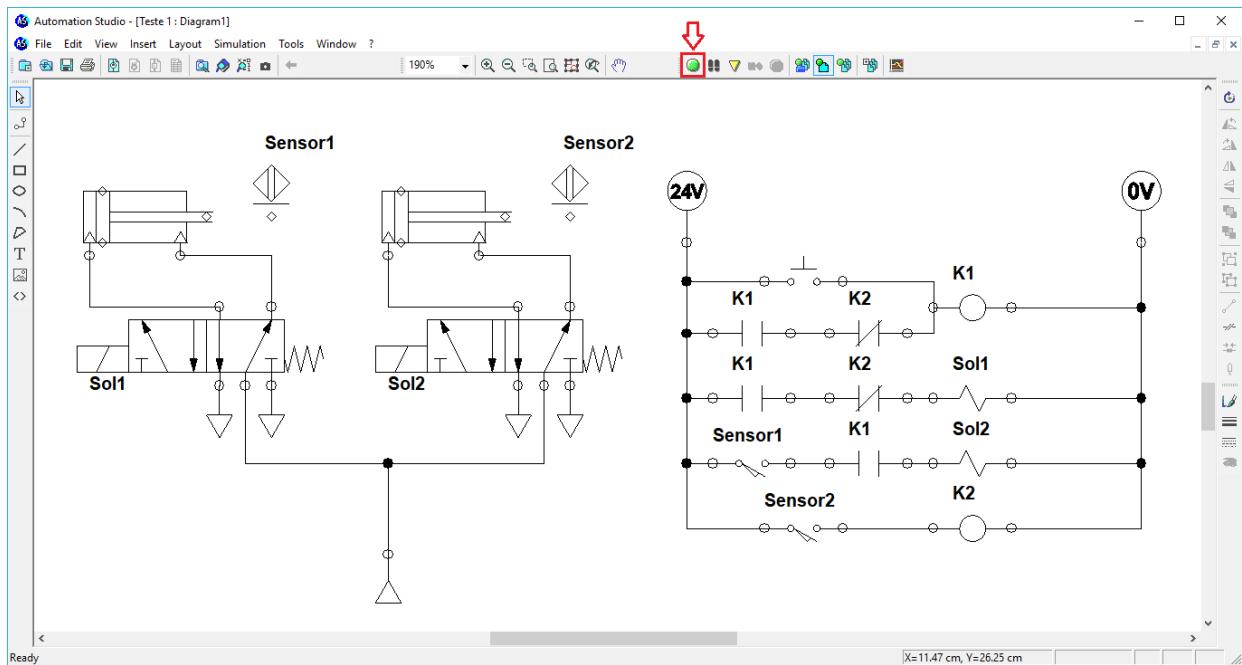


Figura 2.3: Modelagem dos circuitos no *Automation Studio*

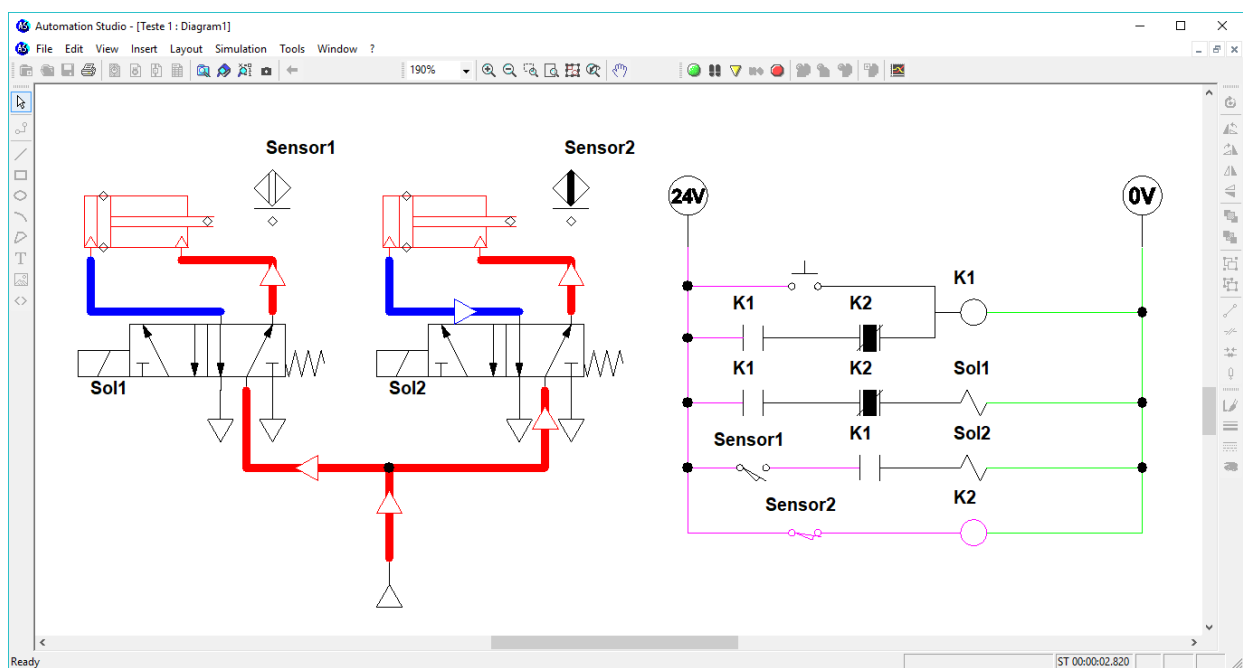


Figura 2.4: Simulação do sistema no *Automation Studio*

Capítulo 3

Estrutura do *software Open FluidSim*

Neste capítulo apresenta-se a estrutura geral do *Open FluidSim*, em alto nível. Inicialmente, define-se o escopo de atuação do simulador. Em seguida, discorre-se acerca do funcionamento dos elementos básicos de sistemas hidráulicos e pneumáticos. Adiante, especifica-se as bibliotecas de elementos que serão implementadas no *software*. Posteriormente, definem-se uma série de requisitos para o simulador, apresentados em formato de histórias de usuário. Em seguida, discorre-se acerca da estrutura geral do *Open FluidSim*. Finalmente, apresenta-se a interface gráfica do simulador.

3.1 Definição do escopo de atuação do software

Conforme citado na Seção 1.4, o *Open FluidSim* visa facilitar o aprendizado de sistemas fluido mecânicos, e não a competição com ferramentas comerciais. Assim, suas funcionalidades são reduzidas quando comparado às ferramentas mencionadas no Capítulo 2. Considerando também as necessidades específicas dos alunos do curso de Sistemas Hidráulicos e Pneumáticos [11] da UnB, definiu-se que o *Open FluidSim* permitirá a simulação de sistemas eletro-hidráulicos e eletro-pneumáticos. Ademais, os elementos elétricos, hidráulicos e pneumáticos serão representados de acordo com a simbologia internacional definida pela norma *Joint Industrial Council*, ou JIC [23].

De acordo com os objetivos do presente trabalho, estabelecidos na Seção 1.4, o *Open FluidSim* deverá ser uma ferramenta multiplataforma. Dificuldades de acesso restringem a quantidade de plataformas que podem ser abordadas neste trabalho. Ademais, nem todas as plataformas possíveis são de interesse particular no presente momento, tais como *Facebook*, máquinas específicas de *video games* e dispositivos de realidade virtual. Portanto, optou-se por desenvolver o *software* para as plataformas *Windows*, *web* e *Android*.

3.2 Elementos básicos de sistemas hidráulicos e pneumáticos

Antes de adentrar na estrutura do *Open FluidSim* é necessário possuir conhecimento do funcionamento básico dos elementos hidráulicos e pneumáticos abordados. Nesta seção discorre-se

acerca de tais elementos e seu funcionamento.

3.2.1 Bombas hidráulicas e reservatórios

Uma bomba hidráulica é um dispositivo utilizado para bombear líquidos de um ponto a outro. Como sistemas hidráulicos utilizam fluidos incompressíveis, a pressão do sistema pode ser aumentada rapidamente conforme necessário. Assim, é comum que reservatórios de água sejam mantidos a pressão atmosférica [1].

Existem diversos tipos diferentes de bombas hidráulicas. Por simplicidade, este trabalho utilizará apenas bombas de deslocamento fixo, cuja simbologia é apresentada na Figura 3.1a, e reservatórios abertos à atmosfera, cuja simbologia é apresentada na Figura 3.1b.

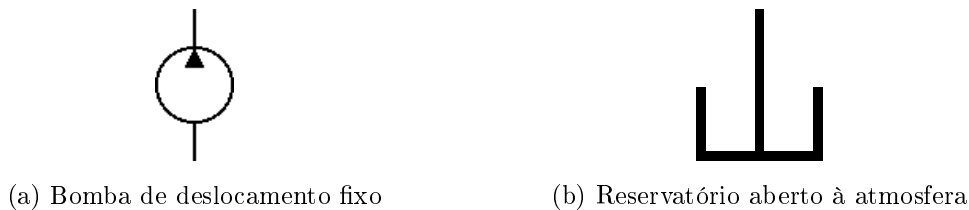


Figura 3.1: Simbologia de bombas e reservatórios

3.2.2 Compressores e exaustores

A maior parte dos sistemas pneumáticos utilizam ar atmosférico comprimido. Dessa forma, o sistema pode ser aberto: o ar é obtido da atmosfera através dos compressores, utilizado e devolvido pelos exaustores.

Sistemas pneumáticos costumam apresentar uma série de elementos acessórios. Como o fluido utilizado é compressível, a pressão do sistema leva mais tempo para atingir níveis operacionais quando comparado a sistemas hidráulicos. Portanto, costumam apresentar reservatórios de ar comprimido, que são mantidos na pressão requerida pelo sistema. Além disso, a compressão do ar aumenta sua temperatura, então uma unidade de resfriamento é adicionada ao sistema. Finalmente, o ar atmosférico precisa ser filtrado antes de poder ser utilizado [1].

Software de simulação, como o *Automation Studio* e o *Simscape Fluids*, da *MathWorks* apresentam uma versão ideal de compressores de ar: a fonte de pressão pneumática, exibida na Figura 3.2a. Esse elemento mantém uma diferença de pressão especificada independentemente de quaisquer valores de fluxo de entrada [24].

Neste trabalho os elementos acessórios de sistemas hidráulicos não serão abordados, apenas a fonte de pressão pneumática.

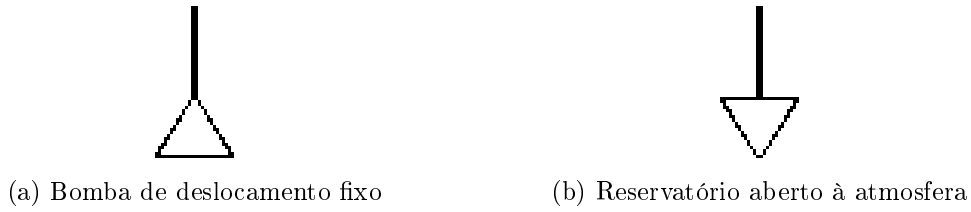


Figura 3.2: Simbologia de bombas e reservatórios

3.2.3 Atuadores lineares

Atuadores são dispositivos capazes de mover ou aplicar força a objetos, e, em geral, classificam-se em lineares ou rotativos. Neste trabalho, apenas os atuadores lineares serão abordados.

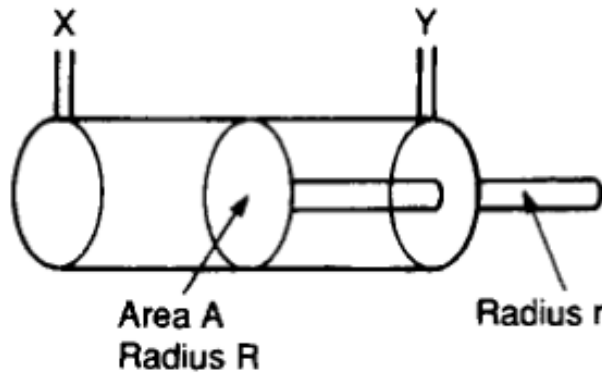


Figura 3.3: Esquema simplificado de um cilindro [1]

A Figura 3.3 apresenta o atuador linear mais básico, o cilindro. Este consiste de um pistão de raio R que se move dentro de um cilindro com entradas de fluidos nas extremidades. O pistão é ligado a uma haste de raio r . A força aplicada pelo pistão depende de sua área e da pressão de entrada. Para o movimento de extensão do pistão, a área A é dada por πR^2 . Se uma pressão P for aplicada no terminal X enquanto o terminal Y estiver aberto, a máxima força de extensão disponível será dada pela Equação 3.1.

$$F_c = P\pi R^2 \quad (3.1)$$

Se, por outro lado, uma pressão P for aplicada no terminal Y enquanto X estiver aberto, o pistão retrai. Nesse caso, sua área total A é reduzida em decorrência da presença da haste de raio r . Portanto, a força máxima de retração será menor que a força máxima de extensão, e é dada pela Equação 3.2. [1]

$$F_c = P\pi(R^2 - r^2) \quad (3.2)$$

O cilindro da Figura 3.3 é classificado como sendo de dupla ação, pois fluidos sob pressão são utilizados tanto para estendê-lo quanto para retrai-lo. Existem outros tipos de cilindro, como o de simples ação, mostrado na Figura 3.4. Neste, o retorno é acionado por uma mola interna.

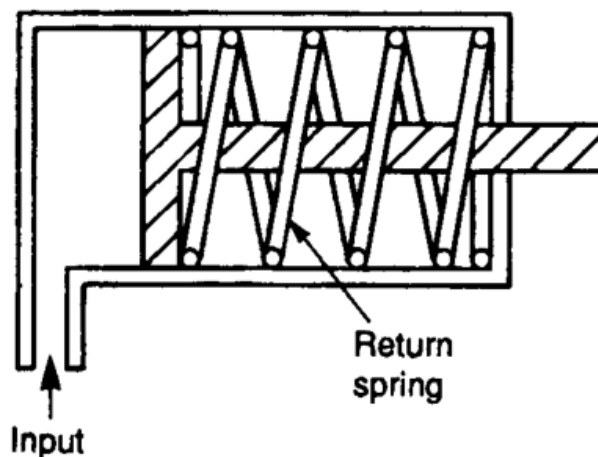


Figura 3.4: Cilindro de simples ação com retorno por mola [1]

3.2.4 Válvulas direcionais

Válvulas são dispositivos que manipulam o fluxo de fluido de diversas formas. Existem 4 tipos básicos de válvulas, tanto para sistemas hidráulicos quanto para pneumáticos. Estas são: direcionais, de bloqueio, de controle de fluxo, e de pressão [10]. Por motivos de escopo, este trabalho aborda apenas as válvulas direcionais.

Válvulas direcionais são mecanismos capazes de alterar o sentido de fluxo dos fluidos em um sistema, ao alterar suas conexões internas. A Figura 3.5 exemplifica esse mecanismo. Válvulas direcionais costumam ser utilizadas para comandar início, término e sentido de movimento de atuadores. [1]

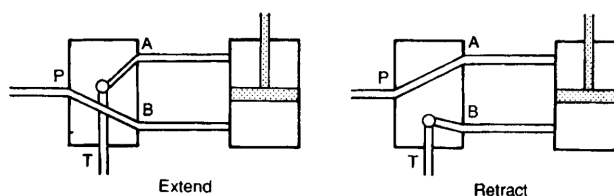
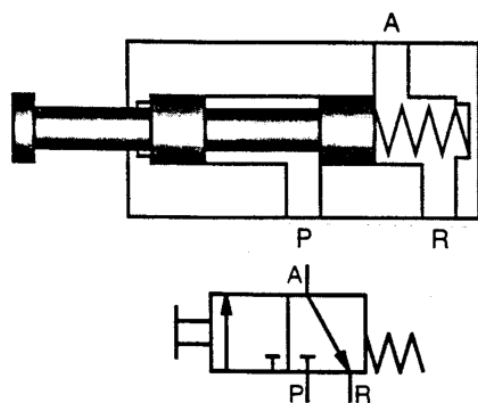


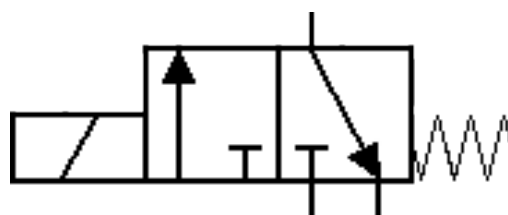
Figura 3.5: Diagrama simplificado de exemplo do funcionamento interno de uma válvula direcional [1]

Válvulas direcionais em geral apresentam uma certa quantidade de terminais e de posições que podem assumir. A nomenclatura dessas válvulas utiliza essas quantidades. Por exemplo, uma válvula com três terminais e duas posições denomina-se “3/2”, enquanto uma outra com quatro terminais e três posições denomina-se “4/3” [1].

Válvulas direcionais possuem formas bastante variadas de acionamento e retorno. A Figura 3.6a apresenta uma visão interna simplificada de uma possível forma de funcionamento de uma válvula 3/2 acionada por botão e com retorno por mola. Um outro método de acionamento é por meio de solenoide, um dispositivo eletromagnético que, aplicado a válvulas direcionais, possibilita o controle em sistema elétrico de seu acionamento. A Figura 3.6b exibe a simbologia de uma válvula 3/2 acionada por solenoide.



(a) Possível funcionamento de uma válvula 3/2 acionada por botão e com retorno por mola [10]



(b) Símbolo de uma válvula 3/2 com acionamento por solenoide e retorno por mola

Figura 3.6: Simbologia de contadores e contatos

3.2.5 Controle elétrico de sistemas fluido mecânicos

Apesar de ser possível projetar e implementar sistemas complexos puramente hidráulicos ou pneumáticos, estes são frequentemente unidos a circuitos elétricos que controlam seu acionamento, transformando-se em sistemas eletro-pneumáticos e eletro-hidráulicos. O controle elétrico abordado na disciplina de Sistemas Hidráulicos e Pneumáticos da Universidade de Brasília baseia-se no uso de solenoides, sensores e contadores. A simbologia adotada para representar graficamente os dispositivos elétricos segue o padrão *Joint Industrial Council*, ou JIC [23].

O contador (ou relé) é um dispositivo eletromagnético composto por uma bobina e um ou mais contatos, sendo alguns fixos e outros móveis. Quando uma corrente elétrica passa pela bobina, um campo magnético é gerado e faz com que os contatos móveis se desloquem. Esse movimento efetua a troca de estado dos contatos, ou seja, os contatos normalmente abertos se fecham, e os normalmente fechados se abrem. A Figura 3.7 apresenta o diagrama de um contador [2], enquanto que a Figura 3.8 apresenta sua simbologia, de acordo com o padrão JIC.

Existem diversos tipos de sensores, cada qual com seu mecanismo de acionamento. Para sistemas fluido mecânicos, é frequentemente desejável detectar a posição dos atuadores, o que pode ser feito por meio de sensores de proximidade [25] [26] ou de fim de curso [27]. Independente do método utilizado, sua representação permanece nos circuitos. De forma similar aos contadores, os sensores possuem contatos normalmente abertos e fechados, além do símbolo do sensor em questão. Sua ativação troca o estado de seus contatos. A Figura 3.9 apresenta a simbologia, seguindo padrão

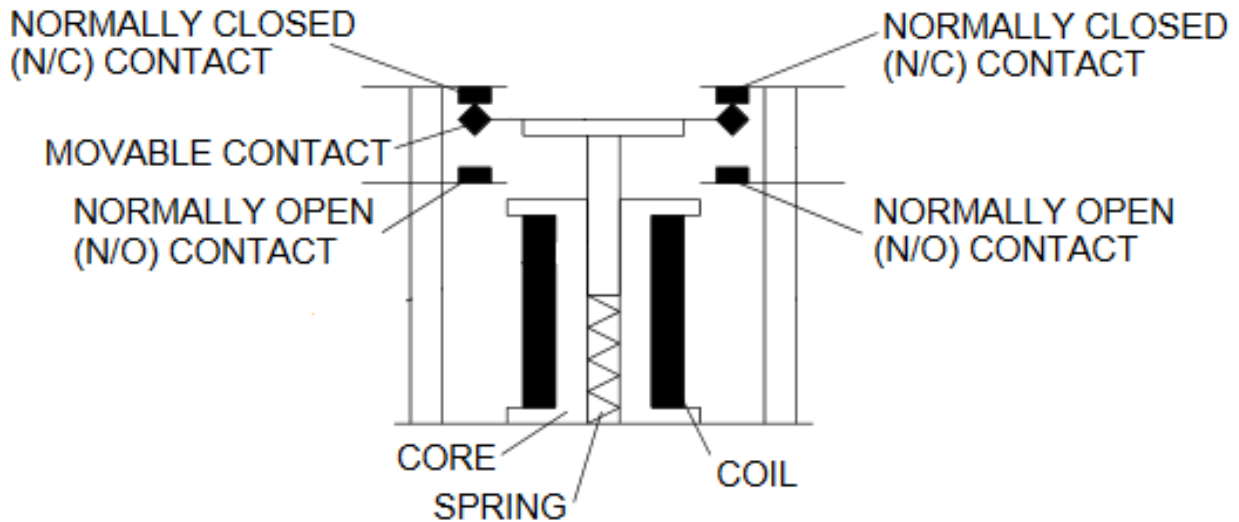


Figura 3.7: Diagrama de um contator [2]

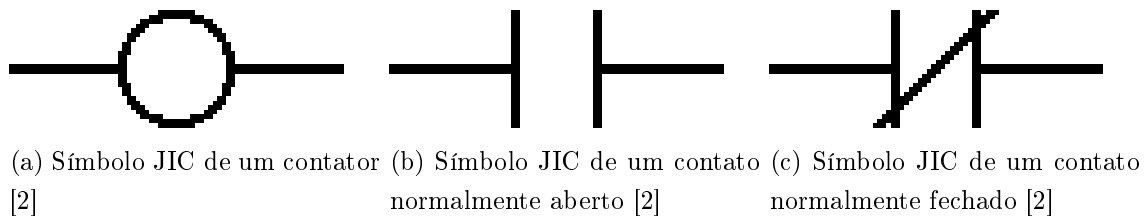


Figura 3.8: Simbologia de contadores e contatos

JIC, dos contatos de sensores, e também de um tipo de sensor de proximidade.

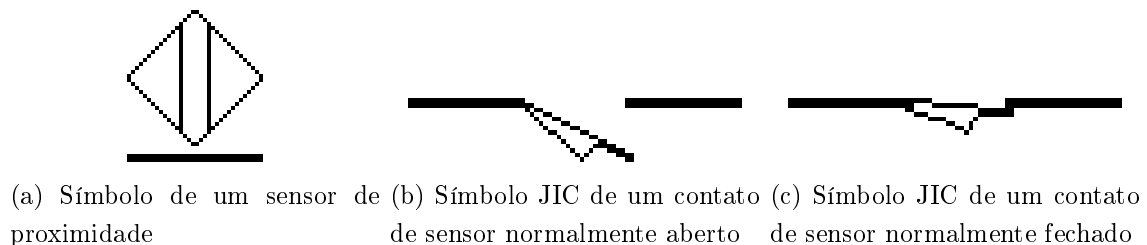


Figura 3.9: Simbologia de sensores

3.3 Especificação da biblioteca de elementos

Conforme citado na Seção 3.1, o simulador visa facilitar o aprendizado de sistemas fluido mecânicos, e não a competição com soluções de simulação comerciais. Dessa forma, a lista de elementos a serem implementados é reduzida, contemplando somente o básico necessário para a implementação dos circuitos estudados na disciplina de Sistemas Hidráulicos e Pneumáticos da Universidade de Brasília [11].



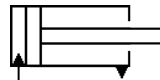
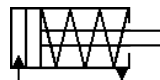
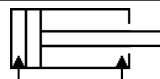
Para sistemas hidráulicos, a biblioteca básica deve incluir um reservatório aberto à atmosfera, elemento de onde líquido é retirado e devolvido, e pelo menos um tipo de bomba hidráulica. Optou-se por incluir apenas uma bomba simples de deslocamento fixo. Além disso, necessita de uma seleção de válvulas de controle e atuadores. Para manter o escopo deste projeto o menor possível, optou-se por implementar apenas quatro tipos de válvulas direcionais, com acionamento por solenoide e retorno por mola, e três tipos de atuadores lineares.

Para sistemas pneumáticos, a biblioteca deve incluir uma fonte de pressão pneumática e um exaustor. Por simplicidade, as válvulas e atuadores pneumáticos escolhidos são similares àqueles da biblioteca hidráulica. Optou-se por não incluir nenhum tipo de compressor de ar, pois a fonte de pressão pneumática representa um compressor ideal que mantém uma diferença de pressão especificada.

Por fim, a biblioteca elétrica deve incluir a quantidade mínima de elementos para possibilitar a implementação de diferentes controladores para os sistemas fluidos. Dessa forma, optou-se por incluir o contator, com seus contatos, solenoide, botões simples, contatos de sensores e fontes de tensão. Esta última tem valores fixos em 24 e 0 Volts. Todos os contatos possuem versões normalmente abertas e fechadas. Os contadores poderão apresentar ativação temporizada.

Os elementos selecionados para implementação estão contidos nas Tabelas 3.1, 3.2, 3.3 e 3.4, bem como os símbolos utilizados no simulador. Para evitar repetições, a tabela 3.3 lista os elementos que fazem parte tanto da biblioteca pneumática quanto da hidráulica. Estes possuem funcionamento e símbolos iguais para ambas as bibliotecas.

Tabela 3.1: Tabela de elementos da biblioteca hidráulica

Denominação	Símbolo
Reservatório aberto à atmosfera	
Bomba simples de deslocamento fixo	
Cilindro de simples ação sem retorno	
Cilindro de simples ação com retorno por mola	
Cilindro de dupla ação	

3.4 Requisitos do software

Depois de especificadas as listas de elementos que irão compor as bibliotecas elétrica, hidráulica e pneumática, deseja-se elaborar uma lista de requisitos para nortear o desenvolvimento do simulador. Tendo em vista os objetivos delimitados na Seção 1.4, e as funcionalidades usualmente

Tabela 3.2: Tabela de elementos da biblioteca pneumática




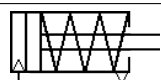

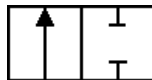

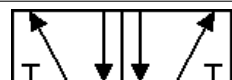
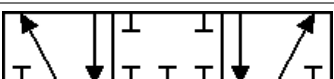

Denominação	Símbolo
Fonte de pressão pneumática	
Exaustor	
Cilindro de simples ação sem retorno	
Cilindro de simples ação com retorno por mola	
Cilindro de dupla ação	





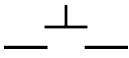
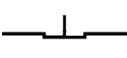
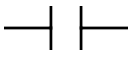
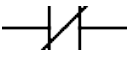
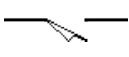

Tabela 3.3: Tabela de elementos compartilhados entre as bibliotecas hidráulica e pneumática

Denominação	Símbolo
Válvula direcional 2/2	
Válvula direcional 3/2	
Válvula direcional 3/2	
Válvula direcional 5/3	
Sensor de proximidade	

apresentadas por *softwares* do gênero, elaborou-se uma lista de requisitos aqui apresentados em formato de histórias de usuário.

1. Como usuário, eu posso visualizar os elementos elétricos, pneumáticos e hidráulicos disponíveis, bem como seus nomes, em uma lista intuitiva;
2. Como usuário, eu posso adicionar, mover e excluir elementos elétricos, pneumáticos e hidráulicos do circuito atual;
3. Como usuário, eu posso selecionar um elemento ao clicá-lo, ou vários elementos ao clicar em uma área vazia e arrastar o *mouse*;
4. Como usuário, eu posso copiar e colar um ou mais elementos;
5. Como usuário, eu posso adicionar conexões entre os elementos presentes no circuito atual;

Tabela 3.4: Tabela de elementos da biblioteca elétrica

Denominação	Símbolo	Denominação	Símbolo
Fonte de tensão fixada em 24 Volts		Terra	
Solenoide		Contator	
Botão normalmente aberto		Botão normalmente fechado	
Contato normalmente aberto		Contato normalmente fechado	
Contato de sensor normalmente aberto		Contato de sensor normalmente fechado	

6. Como usuário, eu posso definir correlações de solenoides, sensores e contatos;
7. Como usuário, eu posso alterar os parâmetros internos de elementos específicos com um duplo clique no elemento.
8. Como usuário, eu posso desfazer e refazer todas as minhas edições do arquivo atual;
9. Como usuário, eu posso salvar o circuito atual em arquivo, e carregá-lo novamente no futuro;
10. Como usuário, eu posso simular o circuito atual para visualizar seu comportamento;
11. Como usuário, eu posso limpar todos os elementos da tela juntamente com o histórico de edições, iniciando um novo arquivo de simulação.

3.5 Estrutura geral do software

O simulador possui dois modos principais, o de edição e o de simulação. A seleção do modo ativo, bem como as funcionalidades individuais e as compartilhadas entre modos são expostas ao usuário pela interface gráfica. O diagrama geral do simulador, contido na Figura 3.10, fornece uma visão de alto nível da hierarquia dos módulos mais relevantes do *software*.

A interface gráfica possui funções de movimentação e alteração do tamanho (*zoom*) da câmera. Também apresenta botões de acesso às funções do editor e simulador em formato de barra de tarefas. Além disso, é responsável por exibir quaisquer mensagens de erro e avisos levantados pelos outros módulos. Finalmente, ela providencia o acesso às bibliotecas de elementos.

O módulo do editor de circuitos apresenta as funcionalidades de edição suportadas pelo simulador. Estas são divididas em uma série de submódulos, conforme o diagrama da Figura 3.10: a ferramenta de salvar e carregar circuitos, o painel de edição, o histórico de edições, o módulo de processamento de entradas de usuário, e a ferramenta de validação de circuitos.

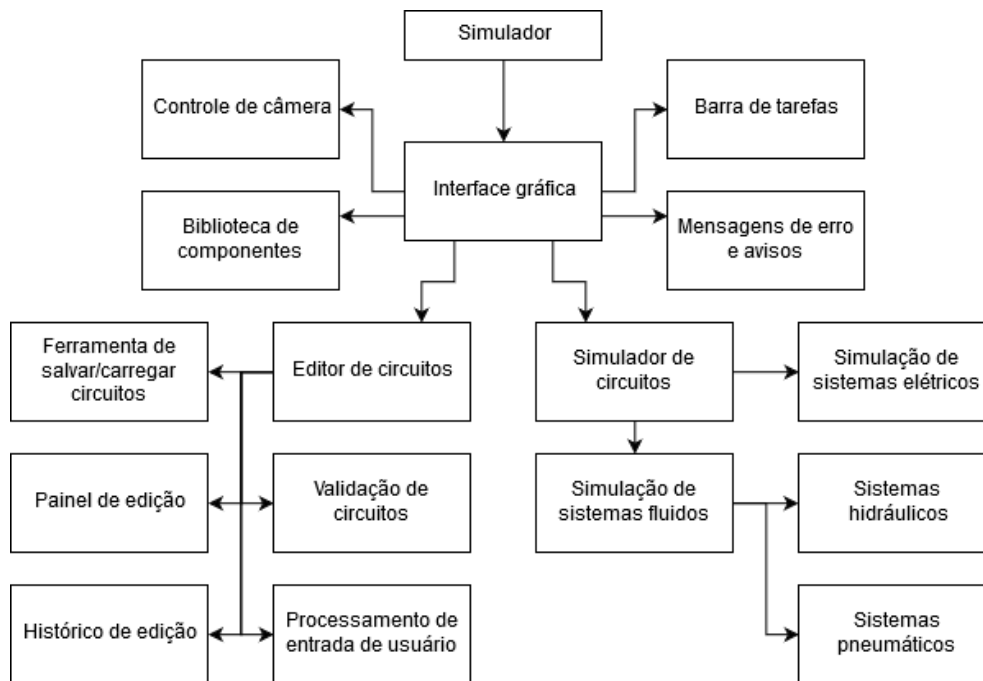


Figura 3.10: Diagrama do *design* de alto nível do simulador

A ferramenta de salvar e carregar circuitos é responsável por converter a informação do circuito atual em um arquivo de texto, e realizar o processo inverso. No caso do carregamento de circuitos de um arquivo, faz-se necessário validá-lo previamente, e caso ocorra algum problema na operação, informar o usuário através do sistema de mensagens da interface gráfica.

O painel de edição constitui o local que contém os elementos e fios do arquivo de circuito atual. Sua função é possibilitar aos outros módulos a visualização e acesso dos objetos presentes no circuito atual. Os elementos são adicionados por meio das bibliotecas da interface gráfica.

Cada edição ao circuito atual realizada pelo usuário configura uma ação que pode ser desfeita e refeita. É função do módulo de histórico de edição conter a pilha de ações, fornecendo aos outros módulos uma interface para a adição de ações ao histórico. Optou-se por não limitar a quantidade de ações que podem ser desfeitas ou refeitas. Limpar o arquivo atual de simulação (excluir todos os elementos adicionados e começar com um arquivo em branco), ou carregar

O módulo de processamento de entrada de usuário implementa a identificação de tipos de clique (simples ou duplo), e fornece ao usuário uma série de atalhos de teclado. Essa estrutura centralizada de processamento de *inputs* fornece também uma maneira simples de desabilitar comandos de edição enquanto o simulador está em modo de simulação.

Antes de sair do modo de editor para iniciar a simulação do circuito é necessário validar os diagramas do arquivo atual. Esta é a tarefa do módulo de validação de circuitos. Se houver elementos não conectados ou se existir algum contato ou solenoide sem correlações uma mensagem de erro correspondente é exibida ao usuário, e a troca de modo para simulação é cancelada.

O módulo do simulador de circuitos contém as *engines* de simulação, que são módulos que implementam os algoritmos utilizados na simulação dos sistemas elétrico, pneumático e hidráulico.

No modo de simulação o usuário apenas observa o comportamento do circuito implementado. Como os únicos elementos dentre aqueles especificados na Seção 3.3 que possibilitam a interação do usuário com o sistema são os botões, o módulo de simulação é comparativamente menor que o de edição.

3.6 Apresentação da interface gráfica

Tendo em vista o objetivo de desenvolver um *software* multiplataforma de edição e simulação de sistemas fluido mecânicos, optou-se pela criação de uma interface gráfica unificada. Em outras palavras, o usuário que utilizar o simulador em um dispositivo *Android* deparará com a mesma interface gráfica da versão para *desktop*.

A Figura 3.11 apresenta uma visão geral do modo de editor de circuitos do simulador. Nesse ponto, a interface divide-se em cinco partes:

1. Barra de bibliotecas de elementos;
2. Painel de edição de circuitos;
3. Botão de opções de controle da câmera;
4. Espaço de mensagens de aviso;
5. Barra de tarefas, com as opções de edição disponíveis;

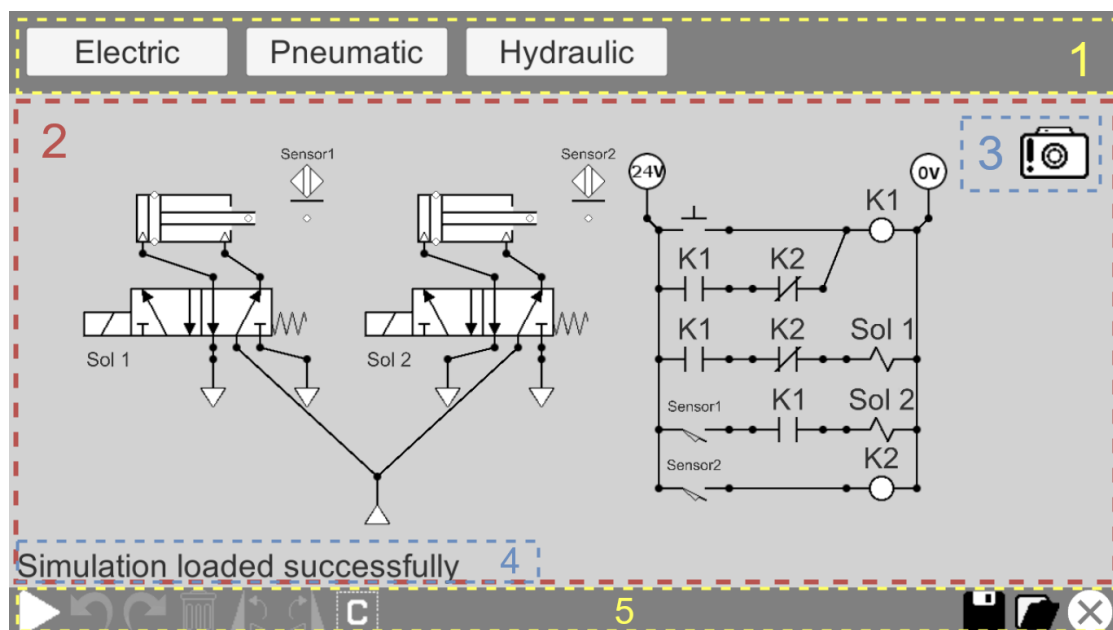


Figura 3.11: Interface gráfica do simulador, visão do modo de editor

3.6.1 Bibliotecas de elementos

As bibliotecas de elementos são acessadas pela barra superior. A Figura 3.12 mostra as três bibliotecas, com as listas de todos os elementos implementados. Cada caixa de elemento é um botão que, quando clicado, fecha o menu de biblioteca e cria um novo elemento flutuante. Este é uma versão temporária e puramente ilustrativa do elemento real, e é utilizado para permitir que o usuário selecione visualmente a posição do elemento antes de adicioná-lo efetivamente ao painel de edição. A Figura 3.12d exemplifica o elemento flutuante.

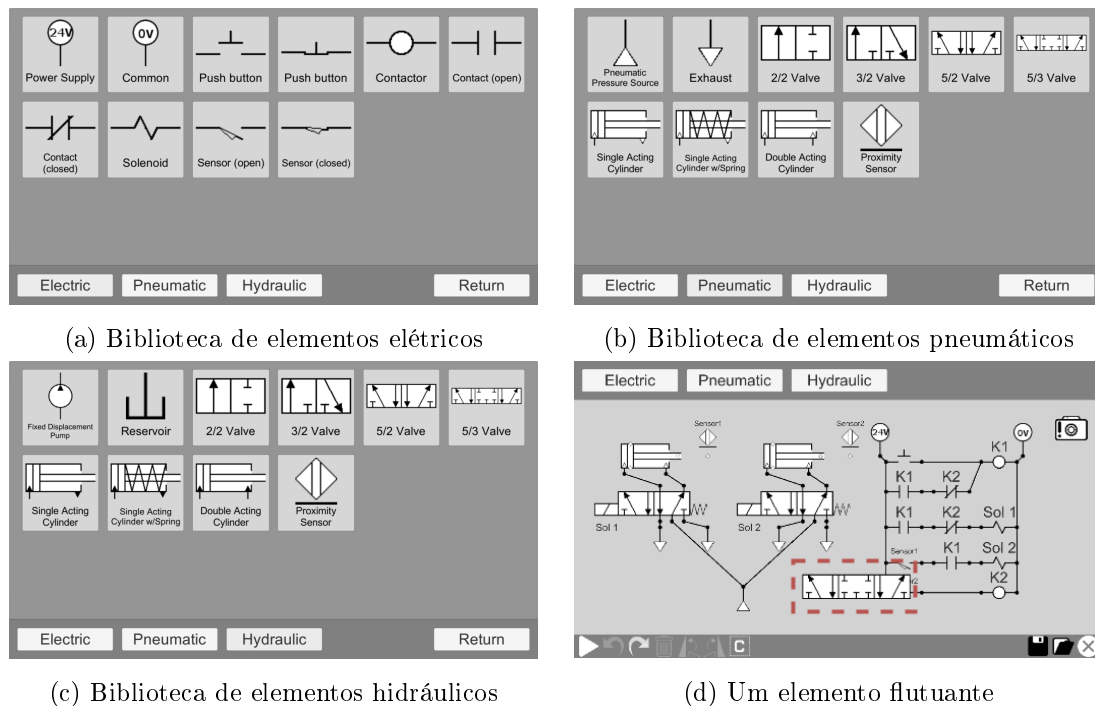


Figura 3.12: Interface das três bibliotecas de elementos, e visão de um elemento flutuante

3.6.2 Painel de edição de circuitos

É no painel de edição de circuitos que o usuário pode modelar os sistemas fluido mecânicos desejados. O conjunto completo de funcionalidades de edição está contido na lista a seguir:

- Adição, movimentação e remoção de elementos;
- Seleção de um ou múltiplos elementos;
- Configuração de cilindros, contadores, contatos e solenoides;
- Estabelecimento de conexões entre terminais de entrada e saída de elementos;
- Comandos de copiar e colar para um ou múltiplos elementos;
- Desfazer e refazer edições.

A Figura 3.13 apresenta os quatro menus de elementos configuráveis. No caso de cilindros, o menu permite a definição da posição inicial do pistão, dada como uma porcentagem do comprimento total, e do tempo em segundos que o pistão leva para deslocar-se ao longo de seu comprimento total. Para contadores, é possível selecionar o tipo de temporização e seu tempo de atraso, dado em segundos. Finalmente, os menus de correlação de contatos e solenoides oferecem a lista de contadores, sensores ou solenoides adicionados ao circuito, para definição de correlações.

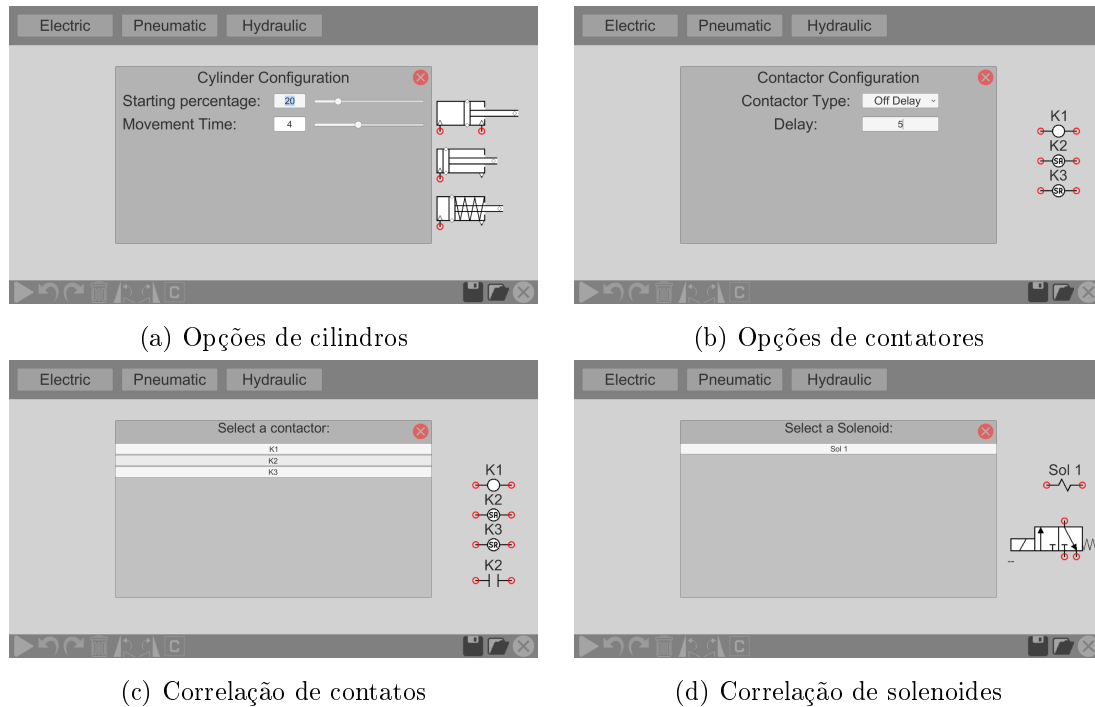


Figura 3.13: Visão das diferentes funcionalidades de edição

Conforme ilustrado nas Figuras 3.13b, 3.13c e 3.13d, alguns elementos possuem um nome identificador. Estes não são definidos pelo usuário. Em vez disso, são gerados pelo simulador sempre que um elemento unicamente identificado é adicionado ao circuito. Existem três tipos de elementos que são unicamente identificados dessa maneira: os contadores, sensores e solenoides.

3.6.3 Modo de simulação de circuitos

Criado o circuito no painel de edição, o usuário pode iniciar a simulação do sistema pelo botão de início, no canto esquerdo da barra de tarefas. As opções de edição são então desativadas e o simulador entra no modo de simulação. A Figura 3.14 apresenta a interface do simulador durante esse modo.

Quando em modo de simulação, os fios e tubulações mudam de cor como forma de exibição visual dos sinais nos conectores. Para o diagrama elétrico, utiliza-se a cor rosa para exibir um valor positivo de sinal, e verde para valores negativos. Em diagramas fluidos as cores escolhidas são vermelho e azul para sinais positivos e negativos, respectivamente. Quando o sinal é nulo, mantém-se a cor preta.

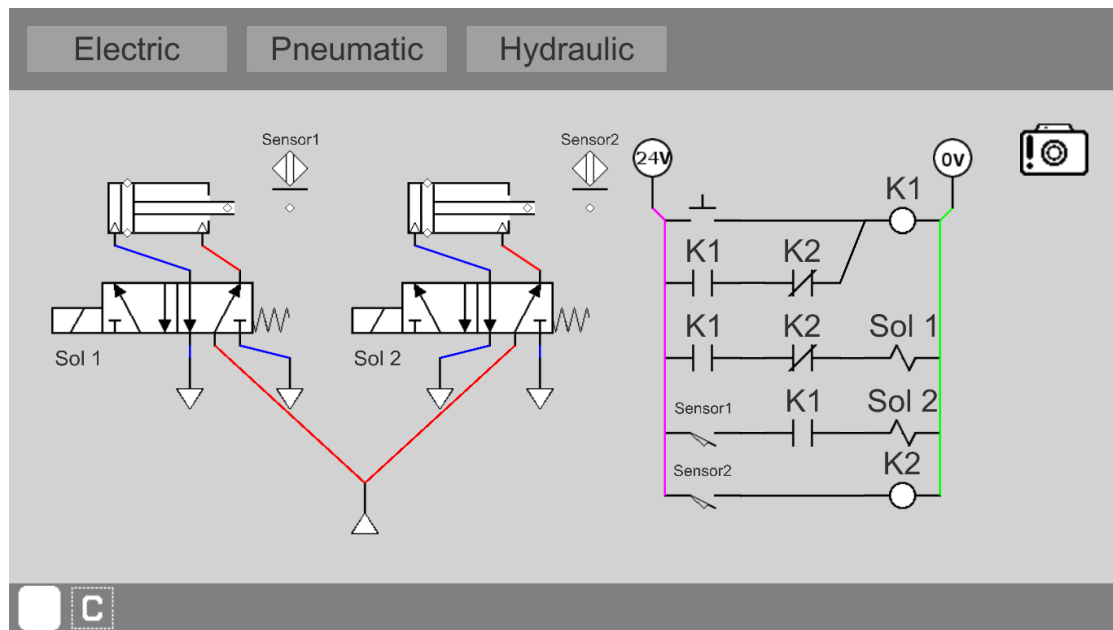


Figura 3.14: Interface gráfica do simulador, visão do modo de editor

Durante o modo de simulação as únicas opções disponíveis ao usuário são as de controle da câmera, e o botão de retornar ao modo de edição. Ademais, os únicos elementos que permitem interação são os dois tipos de botão: normalmente aberto e fechado.

Capítulo 4

Implementação

Neste capítulo discorre-se acerca da implementação da ferramenta *Open FluidSim*. Primeiro, apresenta-se o funcionamento da *Engine Unity*, utilizada para a implementação do simulador. Depois, discute-se o *design* de baixo nível da ferramenta desenvolvida, ou seja, as decisões específicas de implementação do simulador. Finalmente, especificam-se os algoritmos utilizados para a simulação dos elementos hidráulicos, pneumáticos e elétricos.

4.1 A engine Unity

Game engine, ou motor de jogo, é um *software* ou *framework* que fornece ferramentas de uso geral em jogos digitais, visando agilizar o processo de prototipagem e desenvolvimento de tais jogos [28]. Existem diversas *game engines* comerciais disponíveis, cada qual com seu conjunto de funcionalidades oferecidas. A mais utilizada na indústria de jogos atualmente é a *Unity* [29]. Ela possui uma licença de uso pessoal gratuita, e que permite seu uso em aplicações não comerciais [30]. Também é capaz de exportar projetos para diversas plataformas [31].

Por essas e outras razões, optou-se por utilizar a *engine Unity* neste trabalho. Apesar de seu foco no desenvolvimento de jogos eletrônicos, a *Unity* possui um conjunto de funcionalidades de grande valor para aplicações além de *video games*. Mais notavelmente, a capacidade de exportação de projetos para uma vasta quantidade de plataformas diferentes constitui em um fator decisivo na sua seleção para o desenvolvimento do *Open FluidSim*.

4.1.1 Noções básicas de Unity

Em *Unity*, um executável é um conjunto com uma ou mais cenas. Uma cena é uma abstração equivalente a uma fase em um jogo digital, e é composta por vários objetos de jogo, ou *Game Objects* [32]. *Unity* utiliza uma arquitetura orientada a componentes: objetos de jogo não possuem nenhuma funcionalidade padrão. Seus comportamentos são definidos em função dos componentes que possuem.

A Figura 4.1 apresenta a visão geral do editor da *Unity*. A aba central é a de cena, que mostra

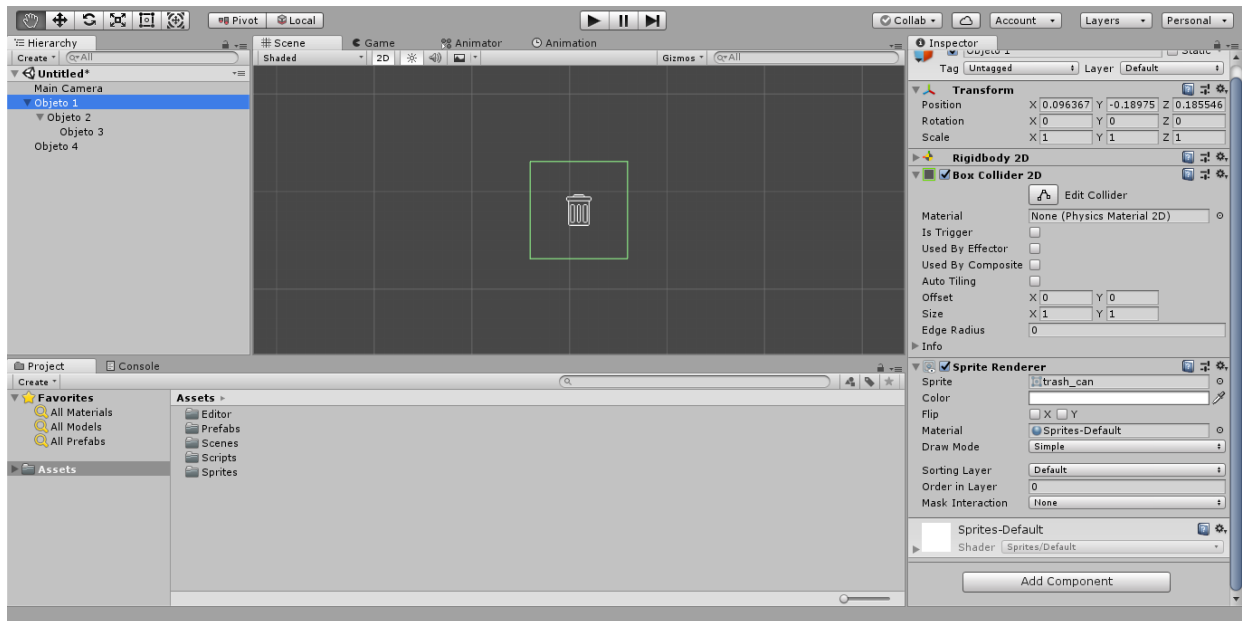


Figura 4.1: O editor *Unity*

visualmente todos os objetos na fase atual. É nesta janela que a cena é organizada visualmente [33].

A aba mais à esquerda fornece a visão da hierarquia dos *Game Objects* na cena. Objetos podem conter outros objetos filhos. Na Figura 4.1, o *Objeto1* é pai do *Objeto2* que, por sua vez, é pai do *Objeto3*. Objetos conservam sua posição relativa a seus pais. Em outras palavras, movimentar, rotacionar ou esticar um *Game Object* faz com que todos os seus objetos filhos se movimentem, rotacionem e estiquem na mesma proporção.

A aba mais à direita é chamada de *Inspector*, e fornece uma visão detalhada de qualquer coisa que estiver selecionada. Na Figura, o inspetor exibe os detalhes do *Objeto1*, que está selecionado. Este possui quatro componentes. O primeiro, denominado *Transform*, mantém a informação de posicionamento, orientação e tamanho do objeto, e é o único componente obrigatório para todo *Game Object*. O segundo denomina-se *Rigidbody2D*, e sua presença confere ao objeto a capacidade de ter seu movimento controlado pela *engine* de física bidimensional da *Unity*. O terceiro componente é o *BoxCollider2D*, utilizado para definir as fronteiras de colisão do objeto. Finalmente, o *SpriteRenderer* é utilizado para a exibição de imagens *bitmap*.

A aba inferior é chamada *Project*, e fornece uma visão dos *Assets* do projeto [34]. Arrastar um objeto da cena para a janela de projeto produz um *Prefab*, uma versão reutilizável do objeto [35]. Sempre que um *Prefab* é alterado, suas instâncias na cena alteram de forma correspondente.

Unity possui um sistema próprio de criação de interfaces gráficas responsivas, e possui implementações dos componentes mais comuns, como botões, barra de rolagem, menu tipo *dropdown*, dentre outros.

A Figura 4.2 ilustra o posicionamento de um botão na cena. A aba de hierarquia mostra que o botão em questão localiza-se dentro de um objeto denominado *Canvas*. Este é utilizado para

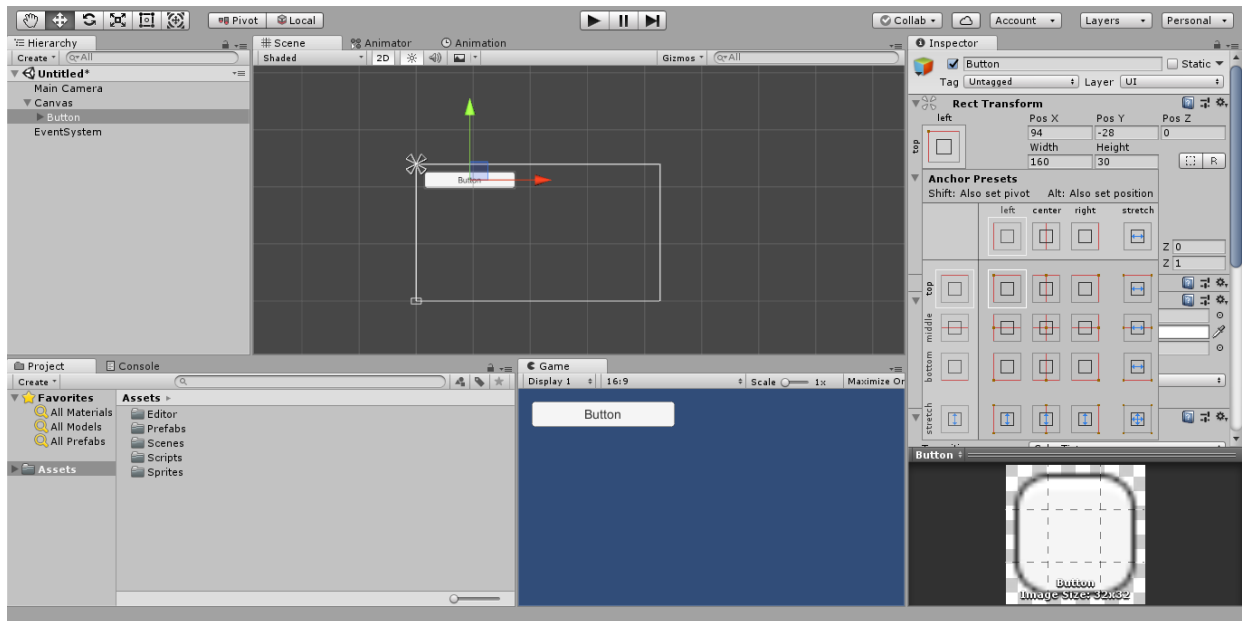


Figura 4.2: Adição de um *UI Button* na tela

definir a área de exibição de elementos da interface gráfica. Todos os objetos que utilizarem o sistema padrão de interfaces da *Unity* devem ser posicionados dentro de algum *Canvas*.

A aba de *Inspector* mostra que o objeto do botão não possui um *Transform* comum: em vez disso, contém um componente denominado *RectTransform*. Este é utilizado apenas nos objetos de interface gráfica da *Unity*. A diferença é que, ao invés de representar um ponto no espaço tridimensional, o *RectTransform* representa um retângulo bidimensional, no plano do *Canvas*. [36]

4.1.2 Scripting em *Unity*

O desenvolvimento em *Unity* consiste basicamente de criar objetos para cada entidade que se deseje implementar, e adicionar a esses objetos componentes que conferem as diferentes funcionalidades desejadas. Entretanto, os componentes padrão da *Unity* são limitados, e feitos para ser de uso geral. Dessa forma, o desenvolvimento de quase qualquer coisa exige que o desenvolvedor crie componentes customizados. Isso é feito por meio de *scripts* [37].

A *Unity* permite o uso de duas linguagens de programação para a criação de *scripts*: *C#* e *JavaScript*. Neste trabalho optou-se por utilizar a linguagem de programação *C#* pois é a mais usada pela comunidade [38]. Portanto, possui mais material disponível na *web*.

A criação de um novo componente requer uma classe que herde da classe *MonoBehaviour*. Feito isso, o *script* pode ser adicionado a um objeto como qualquer outro componente, conforme mostra a Figura 4.3.

A API (Application Programming Interface) de *Scripting* da *Unity* [39] confere aos *scripts* maneiras de controlar praticamente todos os parâmetros de componentes e de configurações de projeto. Além disso, os *scripts* que herdam da classe *MonoBehaviour* têm acesso às funções de

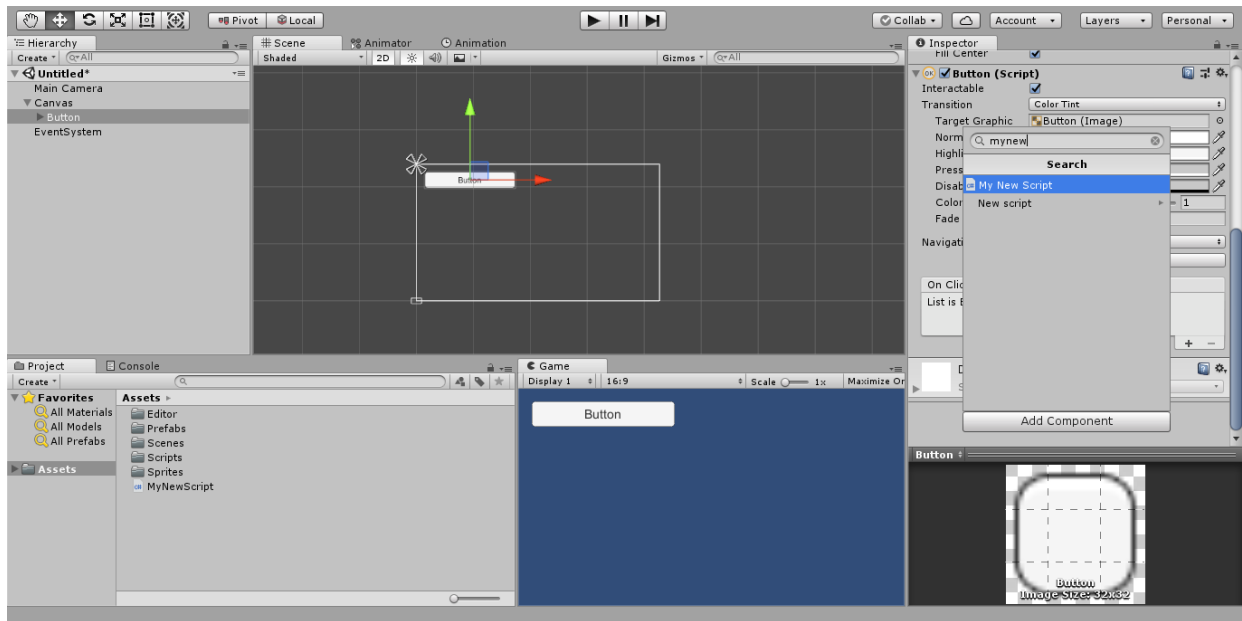


Figura 4.3: Adição de um *script* a um objeto

atualização da *Unity*. Estas são listadas a seguir:

- *Awake*: É chamada uma única vez, imediatamente após a criação da instância do objeto;
- *Start*: É chamada uma única vez, no *frame* seguinte à instanciação do objeto;
- *Update*: É chamada a todo *frame*;
- *LateUpdate*: É chamada no final de cada *frame*;
- *FixedUpdate*: É chamada em intervalos de tempo regulares.

Existem diversas outras funções padrão da *Unity*, mas estas cinco são as mais comuns.

4.1.3 *JSON Utility*

JavaScript Object Notation, ou *JSON*, é um formato textual de intercâmbio de dados baseado na linguagem de programação *JavaScript*. Projetado para ser de fácil leitura tanto para seres humanos como máquinas. Sua representação é independente de qualquer linguagem de programação. Entretanto, mantém convenções familiares para programadores. Um arquivo *JSON* pode apresentar duas estruturas básicas: pares de chave e valor, e listas ordenadas de valores [40].

Unity possui uma ferramenta própria para conversão de objetos para seu equivalente em *JSON*, e vice-versa. Apesar de suportar somente objetos simples, essa ferramenta é mais rápida que outras soluções populares para *.NET* [41].

4.2 *Design* de baixo nível

4.2.1 Modelagem do elementos no *Open FluidSim*

Conforme visto na Seção 2.1, o *Automation Studio* permite a adição de bibliotecas de elementos a partir de arquivos externos. Isto possibilita a adição de novas bibliotecas sem requerer uma atualização de software.

Visando manter o escopo deste trabalho o menor possível, optou-se por modelar cada elemento especificado na Seção 3.3 de forma interna ao *software*. Em outras palavras, a adição de novos elementos ao simulador requer a modelagem destes dentro do ambiente da *Unity*, e a criação de uma nova versão do simulador.

Optou-se por representar cada elemento hidráulico, pneumático e elétrico dentro do simulador utilizando três *prefabs* distintos: o da interface gráfica, o de seleção flutuante e o elemento efetivo. Os dois primeiros são utilizados pela interface gráfica, e constituem passos necessários para selecionar e adicionar elementos ao circuito.

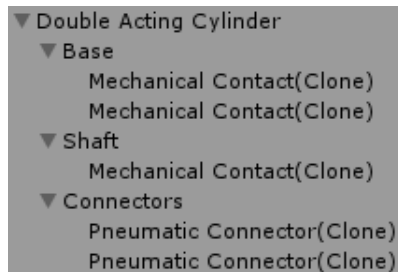
O objeto do elemento efetivo possui suas funcionalidades divididas em diversos componentes diferentes, e ao longo de múltiplos objetos filhos. A relação de componentes utilizados e a hierarquia de objetos filhos varia consideravelmente em função do elemento representado, mas existem alguns requisitos comuns a todos os elementos:

- Um ou mais componentes *Sprite Renderer*, utilizados para exibir os desenhos de elementos, e implementar eventuais animações de simulação;
- Algum tipo de *Collider 2D*, que define a área clicável do elemento;
- Os *scripts* básicos de elementos, que conferem as funcionalidades de edição, além de servirem para identificar o tipo de elemento do objeto;
- Conectores (elétrico, hidráulico e pneumático) e contatos, conforme especificação do elemento;
- Um ou mais *scripts* implementando comportamentos específicos em simulação, como a movimentação de atuadores.

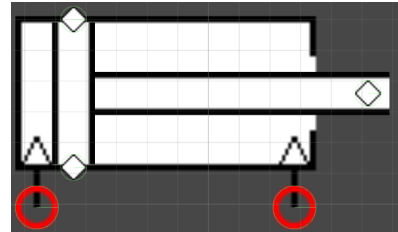
As conexões entre entradas e saídas dos diferentes elementos são efetuadas por meio dos conectores, que podem ser elétricos, hidráulicos e pneumáticos, e constituem objetos que devem ser adicionados à hierarquia do elemento. Os três objetos utilizam uma estrutura similar, e a separação é feita de modo a impedir que o usuário conecte elementos pertencentes a sistemas diferentes, como uma fonte de tensão a um cilindro pneumático.

No caso de atuadores e sensores, optou-se por utilizar “contatos mecânicos”, objetos contidos na hierarquia do elementos e que possuem componentes de colisão. Dessa forma, utiliza-se o módulo de física e detecção de colisões da *Unity* na implementação dos sensores.

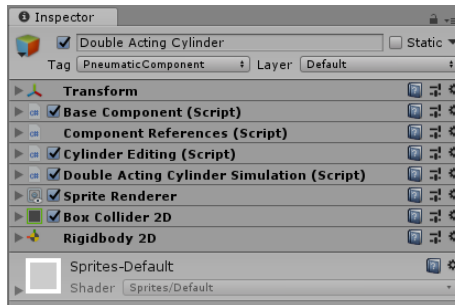
A Figura 4.4 utiliza o cilindro pneumático de dupla ação para exemplificar a representação de um elemento dentro do simulador. A Figura 4.4a mostra o objeto que representa o cilindro, exibindo o posicionamento dos conectores pneumáticos e contatos mecânicos na hierarquia. A Figura 4.4b apresenta o cilindro na janela de cena, com os conectores em vermelho, e os contatos mecânicos em ambos os lados da base e na ponta da haste. Finalmente, a Figura 4.4c mostra a relação dos componentes pertencentes ao objeto do cilindro de dupla ação.



(a) Hierarquia do elemento



(b) Visão do cilindro na janela de cena



(c) Lista dos componentes do elemento, visto no *Inspector*

Figura 4.4: Representação do cilindro pneumático de dupla ação dentro do simulador

4.2.2 Conexões entre elementos e representação de fios e tubulações

Softwares como o *Automation Studio*, que apresentam um editor visual dos sistemas em questão, costumam oferecer a possibilidade de estabelecer conexões entre elementos representados por conjuntos complexos de linhas, representações visuais de fios e tubulações. A Figura 4.5 exemplifica essa funcionalidade, exibindo uma fonte de pressão pneumática conectada a um cilindro de dupla ação através de um fio que passa ao redor do elemento.

Conforme mencionado anteriormente, as conexões entre elementos são efetuadas por meio de conectores, objetos que representam as entradas e saídas de um elemento. O *Game Object* principal que representa um elemento contém uma lista ordenada dos conectores presentes. A ordenação é utilizada na implementação do comportamento em simulação dos elementos, em especial das válvulas direcionais. Os conectores, por sua vez, possuem a lista de todos os outros conectores a ele ligados. Portanto, cada circuito criado no arquivo atual de simulação pode ser entendido como um grafo representando de forma descentralizada, termo aqui utilizado devido ao espalhamento dos nós em múltiplos objetos e componentes.

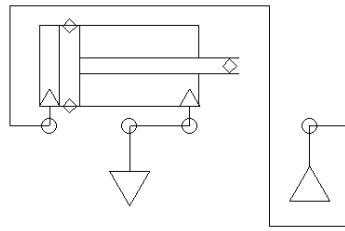
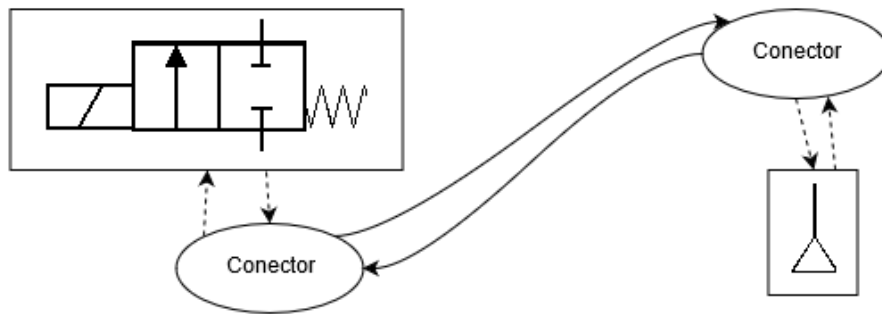
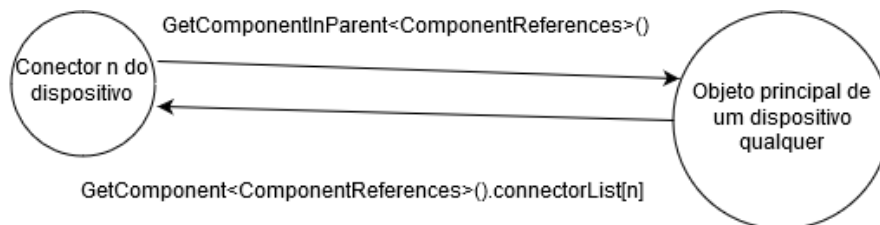


Figura 4.5: Exemplo de conexão entre elementos pneumáticos com tubulação complexa no *Automation Studio*

A Figura 4.6a exemplifica esse grafo de conexões utilizando dois elementos escolhidos arbitrariamente. As linhas contínuas denotam ligações externas, ou seja, entre dois objetos localizados em hierarquias distintas. As linhas pontilhadas denotam ligações internas, ou seja, entre dois objetos na mesma hierarquia. A Figura 4.6b detalha o formato do acesso a um conector, a partir do objeto principal, a nível de código fonte, utilizando as funções da *Unity* de acesso a componentes.



(a) Grafo de conexões entre elementos arbitrários



(b) Detalhamento do acesso de um conector ao objeto principal, e vice-versa

Figura 4.6: Diagramas do funcionamento das conexões entre elementos

A arquitetura adotada para a criação de conexões entre as entradas e saídas de elementos ignora a existência de tubulações e fios. Em outras palavras, assume-se que todos os fios tem comprimento igual a zero. Entretanto, as conexões entre elementos requerem uma representação visual, não só como forma de modelar uma tubulação real, mas também como forma de orientar o usuário do simulador acerca das ligações estabelecidas.

Portanto, a representação de fios e tubulações no simulador desenvolvido é meramente cos-
mética. Por simplicidade, optou-se por representá-las apenas com linhas retas que unem dois conectores. A criação destas é automática, e as linhas se movimentam junto com os elementos

quando necessário. Também são utilizadas no modo de simulação para identificação de sinais e fluxo nos circuitos elétricos e fluidos, mudando de cor conforme o estado da simulação.

4.2.3 Funcionamento da função de salvar e carregar circuitos

A capacidade de salvar e carregar circuitos criados em arquivos é requisito do *software* de simulação. Portanto, é necessário definir o formato do arquivo que irá conter as informações dos circuitos. Optou-se por utilizar o formato *JSON*, descrito na Seção 4.1.3, pois a *Unity* oferece uma ferramenta capaz de realizar a conversão de objetos para *strings JSON*, e vice-versa.

A Listagem 4.1 apresenta uma forma resumida do formato adotado para a representação dos circuitos criados com o simulador em arquivos de texto, utilizando como exemplo uma fonte de pressão pneumática.

O arquivo consiste de uma lista não ordenada de elementos. O primeiro dado salvo é o nome dado ao elemento, que é utilizado para localizar os *prefab* correspondente na biblioteca de elementos. Em seguida, registra-se o número identificador do objeto. Este é criado em tempo de execução pela *Unity*, que garante sua singularidade para cada objeto na cena. Os números identificadores de instância de cada elemento e conector são registrados para que a ferramenta de carregamento de arquivos salvos possa reconstruir as conexões entre componentes, bem como correlações entre elementos (contatos, sensores e solenoides). Continuando, o arquivo apresenta a posição do elemento, e as informações dos seus conectores.

Listagem 4.1: Representação em formato *JSON* de um arquivo contendo apenas uma fonte de pressão pneumática

```
1      {"components": [  
2      {  
3      "name": "Pneumatic Pressure Source",  
4      "componentId": -1312,  
5      "position": { "x": 0, "y": 0, "z": 0 },  
6      "connectors": [  
7      { "connectorId": -1338, "otherConnectorIds": [] }  
8      ],  
9      "solenoids": [],  
10     "contactData": {},  
11     "cylinderData": {},  
12     "contactorData": {},  
13     }  
14     ]}]
```

As outras estruturas salvas são específicas a determinados elementos. A lista de solenoides é utilizada para registrar correlações de solenoides de válvulas. A estrutura de dados de contato é utilizada para guardar a correlação de contatos com suas respectivas contadoras ou sensores. A

estrutura de cilindros é utilizada para registrar seus parâmetros de movimento, e a estrutura de contadores é utilizada para registrar dados de temporização, quando aplicável.

Os módulos de salvar e carregar arquivos de circuitos possuem também um modo de execução parcial. Este é utilizado na implementação da ferramenta de copiar e colar elementos. Neste caso, apenas a seção selecionada do circuito é convertida para a notação textual em *JSON*, e o texto gerado não é salvo em um arquivo externo, mas numa variável de área de transferência interna ao *software* para processamento pelo módulo de carregamento de circuitos conforme solicitação do usuário.

4.2.4 Funcionamento do histórico de ações

A capacidade de desfazer e refazer edições realizadas configura um dos requisitos do simulador. Dessa forma, criou-se um módulo responsável por manter o registro de cada ação realizada durante o uso do editor. As ações de edição que o usuário pode efetuar, e que são salvas no histórico, estão enumeradas a seguir.

1. Adicionar um novo elemento ao circuito;
2. Mover um ou mais elementos;
3. Rotacionar um ou mais elementos em sentido horário ou anti-horário;
4. Estabelecer uma nova ligação entre elementos;
5. Definir uma correlação de contatos de contator ou sensor, ou de solenoides;
6. Colar um ou mais elementos (previamente copiados) no circuito;
7. Excluir um ou mais elementos.

Todas as ações mencionadas são representadas por classes que implementam a interface de ação. Esta define quatro métodos que ações devem implementar: o de realizar a ação, o de desfazê-la, o de refazê-la, o de destruí-la. Além disso, devem especificar um nome, que é exibido ao usuário na forma de mensagem no canto da tela, para que este possa visualizar a denominação das ações quando são desfeitas e refeitas.

Ações permanecem no histórico até que sejam apagadas. Ações desfeitas são transferidas para uma pilha auxiliar. Sempre que uma nova edição é realizada, a pilha auxiliar é esvaziada, e quaisquer ações que nela estiverem são destruídas. Quando o usuário solicita a destruição do arquivo atual de simulação, ou carrega um circuito de um arquivo previamente salvo, todas as ações do histórico são destruídas, junto com quaisquer elementos que estiverem presentes no painel de edição.

4.3 Especificação dos algoritmos de simulação

A simulação dos sistemas hidráulicos e pneumáticos utiliza dois módulos separados de simulação, um para os circuitos elétricos e outro para os fluidos. Além disso, cada elemento implementa seus comportamentos específicos.

Nesta seção apresentam-se os algoritmos especificados para a simulação dos diferentes sistemas implementados.

4.3.1 Simulação dos circuitos elétricos

Quando o modo de simulação é iniciado, o módulo de simulação de sistemas elétricos busca no painel de simulação todos os conectores elétricos ativos. Neste processo identificam-se também aqueles que pertencem às entradas e saídas do diagrama elétrico, ou seja, a fonte de tensão e o terra. Os elementos elétricos ativos no circuito também são notificados do início da simulação, e executam rotinas de inicialização específica conforme necessário. A Listagem 4.2 mostra a função encarregada da inicialização de simulação.

Na função *Update*, chamada automaticamente pela *Unity* a cada *frame*, três passos são executados. Inicialmente, percorre-se por todos os conectores elétricos e define-se o sinal contido como 0. Em seguida, para cada elemento de saída (terra, ou 0 Volts) define-se o sinal de seu conector como -1, e propaga-se esse sinal para todos os outros conectores ligados. A Listagem 4.3 exhibe o funcionamento da função de atualização.

A função de propagação de sinal recebe como parâmetros um conector e o valor do sinal a ser propagado. Se o sinal do conector recebido for diferente de zero, ela encerra, pois este já foi processado. Isso impede que o algoritmo execute indefinidamente. Se o conector não tiver sido processado ainda, define-se seu sinal como o fornecido para a função, e verifica-se a presença de algum *script* que implemente um elemento elétrico. Caso afirmativo, chama-se a sua função de responder a sinais elétricos. Após o envio do sinal ao objeto principal, percorre-se a lista de conectores conectados, e propaga-se o sinal em questão novamente. A Listagem 4.4 mostra o funcionamento da função de propagação dos sinais.

A implementação da função de recebimento do sinal elétrico é particular a cada elemento. Seus parâmetros são o conector fonte (por onde o sinal foi recebido) e o valor do sinal. Contatos de contadores, por exemplo, apenas alternam entre propagar o sinal recebido em um conector para o outro, e assim para outros elementos do circuito, e ignorar o sinal recebido, bloqueando-o. Contatos de sensores e botões possuem o mesmo funcionamento, variando apenas o mecanismo de troca de estado.

Contadores e solenoides, por outro lado, nunca propagam sinais recebidos. Em vez disso, eles são ativados quando, ao final de um *frame*, receberem ambos os sinais, tanto o positivo quanto o negativo, em seus dois conectores, independentemente da ordem e posição de chegada dos sinais.

Listagem 4.2: Função de inicialização da *engine* de simulação de circuitos elétricos

```

1      public void Setup() {
2          connectors = SimulationPanel.instance.GetActiveElectricConnectors();
3          foreach (var connector in connectors) {
4              if (connector.GetComponentInParent<PowerSupplySimulation>())
5                  sourceConnectors.Add(connector);
6              if (connector.GetComponentInParent<ElectricCommonSimulation>())
7                  commonConnectors.Add(connector);
8          }
9          foreach (var component in
                SimulationPanel.instance.GetActiveElectricComponents())
10             if (component.GetComponent<ElectricComponent>())
11                 component.GetComponent<ElectricComponent>().Setup();
12             simulating = true;
13     }

```

Listagem 4.3: Função *Update* da *engine* de simulação de circuitos elétricos

```

1      void Update() {
2          if (simulating) {
3              ClearSignals();
4              foreach (var common in commonConnectors)
5                  SpreadSignal(common, -1f);
6              foreach (var source in sourceConnectors)
7                  SpreadSignal(source, 1f);
8          }
9      }

```

Listagem 4.4: Função de propagação de sinais da *engine* de simulação de circuitos elétricos

```

1      public void SpreadSignal(Connector source, float signal) {
2          if (source.signal == 0) {
3              source.signal = signal;
4              var electricComponent =
                    source.GetComponentInParent<ElectricComponent>();
5              if (electricComponent) electricComponent.RespondToSignal(source,
                    signal);
6              foreach (var other in source.connectedObjects)
7                  if (other.signal == 0)
8                      SpreadSignal(other, signal);
9          }
10     }

```

4.3.2 Simulação dos circuitos fluidos

O modelo adotado para a simulação dos sistemas fluidos contém as simplificações físicas mais significativas deste projeto, e foi criado visando primariamente a simulação dos comportamentos lógicos dos circuitos abordados na disciplina de Sistemas Hidráulicos e Pneumáticos. Aqui, optou-se por abstrair os cálculos efetivos de pressão e fluxo de fluido, em favor de um sistema mais simples. O módulo de simulação de sistemas fluidos abrange tanto os circuitos pneumáticos quanto os hidráulicos.

De forma análoga ao funcionamento do módulo de simulação de circuitos elétricos, a diferença de pressão estabelecida pela bomba hidráulica de deslocamento fixo e a fonte de pressão pneumática introduz no circuito um “sinal” de pressão positivo. Exaustores e reservatórios, por outro lado, introduzem ao sistema sinais de pressão negativos. Estes propagam-se pelo circuito utilizando o mesmo algoritmo apresentado para a propagação de sinais elétricos na *engine* de simulação elétrica.

O comportamento das válvulas direcionais é similar ao de contatos e botões em diagramas elétricos, apenas transmitindo os sinais advindos das entradas e saídas através de suas vias, ou bloqueando-os conforme delimitado na simbologia.

Já nos atuadores, o comportamento dos cilindros de simples ação é razoavelmente trivial. Movimentam-se caso o único conector apresentar sinal de pressão positivo. O cilindro com retorno por mola volta à sua posição inicial apenas quando receber um sinal negativo. Já os cilindros de dupla ação, com dois conectores, deslocam-se no momento em que receberem o sinal positivo e o negativo em seus terminais, em um mesmo *frame*. O deslocamento é realizado no sentido do terminal que apresentar o sinal positivo ao que receber o sinal negativo.

Em decorrência da escolha de utilizar apenas sinais de pressão positivos ou negativos e de valor arbitrário, o cálculo da magnitude da velocidade de deslocamento dos atuadores não pode utilizar os valores de sinal recebidos. Em vez disso, o usuário pode definir diretamente o tempo que um cilindro leva para movimentar-se de uma ponta a outra.

As fontes de pressão pneumática, exaustores e reservatórios são elementos de entrada e saída dos circuitos fluidos, visto que estes apresentam apenas um conector. A bomba hidráulica de deslocamento fixo, no entanto, apresenta dois conectores, e não são consideradas elementos de entrada. Em outras palavras, a *engine* de simulação fluida inicia a propagação de sinal apenas nas fontes de pressão pneumática, nos exaustores e nos reservatórios, conforme mostra a Listagem 4.5. A propagação de sinais positivos de pressão em circuitos hidráulicos é feita pela bomba de deslocamento fixo apenas quando esta recebe um sinal negativo em seu terminal inferior, conforme mostra a Listagem 4.6.

Listagem 4.5: Função de atualização da *engine* de simulação de circuitos fluidos

```
1      void Update() {
2      if (simulating) {
3      ClearSignals();
4      foreach (var exhaust in exhausts)
5      SpreadSignal(exhaust, -1f);
6      foreach (var source in pressureSources)
7      SpreadSignal(source, 1f);
8      foreach (var reservoir in reservoirs)
9      SpreadSignal(reservoir, -1f);
10     }
11 }
```

Listagem 4.6: Função de processamento de sinal da bomba hidráulica

```
1      public class HydraulicPumpSimulation : FluidComponentSimulation {
2      public override void RespondToSignal(Connector sourceConnector,
3      float signal) {
4      if (sourceConnector == componentReferences.connectorList[0]
5      && signal == -1)
6      FluidSimulationEngine.instance.SpreadSignal(
7      componentReferences.connectorList[1], 1f
8      );
9      }
10 }
```

Capítulo 5

Validação do simulador - Testes

O correto funcionamento dos algoritmos de simulação do *Open FluidSim* precisa ser validado. Neste capítulo apresentam-se testes propostos para efetuar tal análise. Inicialmente, apresenta-se a metodologia adotada para implementação dos testes. Em seguida, propõe-se uma série de circuitos de teste, cujo funcionamento em simulação é apresentado utilizando o *software Automation Studio*. Finalmente, apresenta-se os resultados da simulação dos circuitos de teste no *Open FluidSim*, e analisa-se os resultados obtidos.

5.1 Metodologia de implementação dos testes

Cada circuito de teste apresentado no presente capítulo consiste de um sistema eletro-hidráulico ou eletro-pneumático. Todos os circuitos apresentados objetivam o movimento de um ou mais atuadores lineares, em uma sequência predeterminedada, iniciando ao apertar um único botão pulsador. Enquanto que o funcionamento de alguns circuitos pode ser considerado trivial, outros apresentam uma complexidade mais elevada. Para facilitar a compreensão do funcionamento esperado para estes circuitos utiliza-se o diagrama trajeto-passo.

5.1.1 Diagrama trajeto-passo

O *Open FluidSim*, assim como outros simuladores comerciais, é um simulador para eventos sequenciais e discretos, cuja melhor representação é o diagrama trajeto-passo. Tais diagramas são um tipo de gráfico utilizado para especificar a sequência de ações de atuadores fluidos. A Figura 5.1 exemplifica um diagrama trajeto-passo simples com dois cilindros lineares. A primeira linha pontilhada corresponde ao recebimento de algum sinal, advindo de botão ou algum outro mecanismo. Quando este sinal é recebido, apenas o cilindro *A* inicia seu movimento. Quando este alcança o final de seu trajeto, o cilindro *B* inicia seu movimento. Quando ambos estiverem no final de seus trajetos, retornam ao estado original simultaneamente.

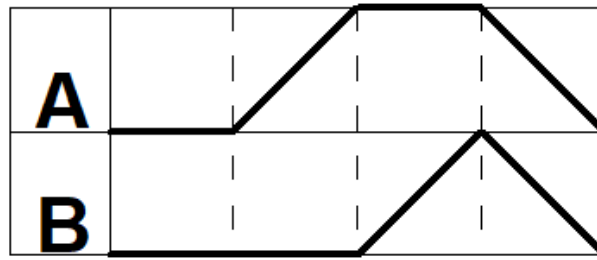


Figura 5.1: Diagrama trajeto-passo de um sistema arbitrário

5.2 Testes Propostos

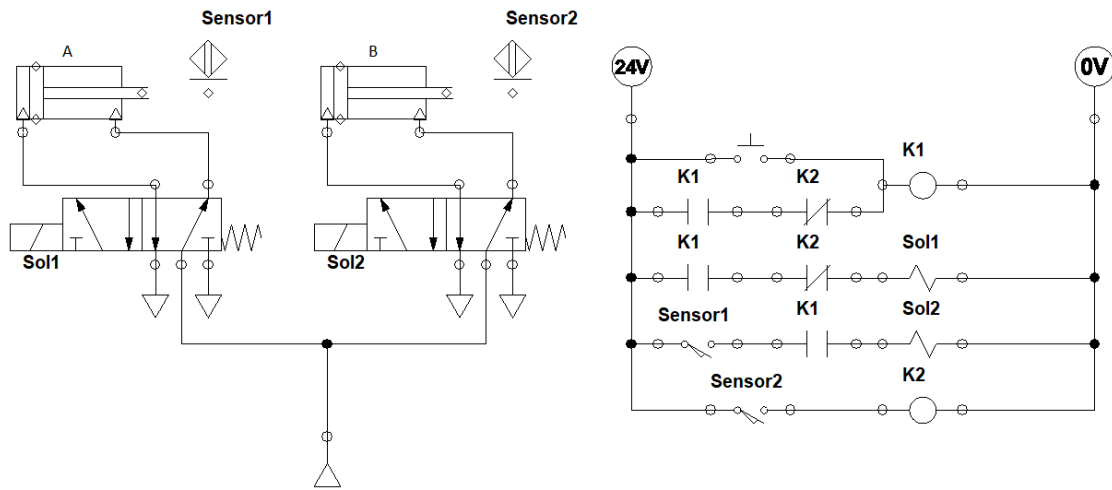
Nesta seção apresentam-se os testes propostos para a avaliação do funcionamento do simulador desenvolvido. Esses testes consistem de uma série de sistemas eletro-hidráulicos e eletro-pneumáticos, construídos utilizando o *Automation Studio* e os elementos selecionados na Seção 3.3. Para cada sistema de teste explica-se o funcionamento esperado dos circuitos e como a simulação ocorre no *Automation Studio*. Diagramas trajeto-passo são utilizados para auxiliar na exposição dos sistemas mais complexos dentre os selecionados.

A Figura 5.2 apresenta o primeiro circuito de teste, um sistema eletro-pneumático com dois atuadores. O diagrama trajeto-passo da Figura 5.2b exibe o comportamento esperado para os dois cilindros. Quando o botão é pressionado, o contator *K1* é energizado, e seus contatos equivalentes, todos normalmente abertos, se fecham, conforme representado na Figura 5.3b. Estes são utilizados tanto para manter seu contator ativo quanto para acionar o solenoide *Sol1*, que provoca o início do movimento do cilindro *A*. Este desloca-se até o fim de seu curso, onde ativa o *Sensor1*, que por sua vez aciona o segundo solenoide, *Sol2*, iniciando o movimento do cilindro *B*, conforme evidenciado na Figura 5.3c. Quando este alcança o final de seu curso, o *Sensor2* aciona o contator *K2*, cujos contatos normalmente fechados abrem os pontos chave do circuito elétrico. Desativados os solenoides *Sol1* e *Sol2*, as válvulas retornam à sua posição inicial, e ambos os cilindros iniciam o movimento de retorno simultaneamente, conforme mostra a Figura 5.3d.

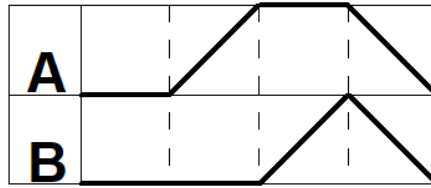
A simulação do circuito da Figura 5.2 no *Automation Studio* se comporta conforme aqui descrito.

A Figura 5.4 apresenta o segundo circuito proposto para teste, novamente um sistema eletro-pneumático. Com apenas um atuador, seu funcionamento é razoavelmente simples, e um diagrama trajeto-passo não é necessário. Ao apertar do botão, o primeiro contator *K1* é acionado, ativando o solenoide *Sol1*, que por sua vez inicia o movimento do cilindro pneumático. Quando este alcança o fim de seu curso, ativa o *Sensor1*, que provoca a troca de estado de seus dois contatos no diagrama elétrico. Neste ponto, o botão antes utilizado para iniciar o movimento pode ser usado para provocar o retorno do cilindro à sua posição inicial.

A simulação desse circuito no *Automation Studio* ocasionalmente apresenta um problema inesperado no diagrama elétrico, onde o contator *K1* oscila entre ativo e inativo. A Figura 5.5 ilustra esse comportamento através de uma sequência consecutiva de dois *frames*. Entende-se que esta é



(a) Diagramas pneumático e elétrico



(b) Diagrama trajeto-passo

Figura 5.2: Diagramas do primeiro sistema de teste

uma falha do algoritmo de simulação de circuitos elétricos do *Automation Studio*.

O terceiro sistema proposto para teste, ilustrado na Figura 5.6, consiste em uma variação do teste anterior. Enquanto que aquele permitia o acionamento do botão durante o trajeto de retorno do cilindro, este garante seu completo retorno antes que o botão possa ser utilizado novamente. A simulação desse sistema no *Automation Studio* apresenta o comportamento normal esperado.

A Figura 5.7 apresenta o quarto circuito de teste proposto, o mais complexo e também o último sistema eletro-pneumático a ser testado. Seu diagrama trajeto-passo ilustrado na Figura 5.7b mostra o comportamento desejado dos dois atuadores. Ao apertar o botão pulsador, o cilindro *A* movimenta-se até a metade, aproximadamente, de seu trajeto completo, onde é interrompido pelo acionamento do *Sensor1*. Este, por sua vez, aciona o movimento do atuador *B*, que percorre seu trajeto completo e ativa o *Sensor2*, fazendo o cilindro *A* retomar seu movimento. No momento em que este alcançar seu fim de curso, o *Sensor3* comanda o retorno do cilindro *B*. Quando este completa seu movimento, o *Sensor4* aciona o retorno do cilindro *A*. Finalmente, o *Sensor5* desativa os contadores e solenoides restantes, e o sistema retorna a seu ponto inicial.

Além do circuito elétrico responsável por controlar o sistema conforme especificado pelo diagrama trajeto-passo, este circuito de teste apresenta também uma estrutura de botão de emergência, utilizando os elementos selecionados para este trabalho. Quando esse botão é pressionado, a alimentação ao circuito principal é cortada, e o solenoide *Sol3* é ativado. Isto é feito de modo a garantir o retorno do sistema a sua posição inicial. O diagrama inclui também um botão normalmente fechado após o de emergência, para que o circuito normal possa ser ativado novamente.

O quinto sistema proposto para testes, apresentado na Figura 5.8, ilustra circuitos eletro-hidráulicos não complexos, visando demonstrar rapidamente o funcionamento dos cilindros de simples ação especificados na Seção 3.3. Ao apertar do botão no diagrama elétrico, os atuadores iniciam seu movimento. Ao soltar o botão, apenas o cilindro com retorno por mola volta à sua posição inicial. A simulação desse circuito no *Automation Studio* apresenta funcionamento normal.

O último circuito de teste, contido na Figura 5.9, apresenta dois atuadores hidráulicos conectados em série, com a saída do cilindro *A* ligada à entrada do *B*. Quando o botão é pressionado, o primeiro atuador inicia seu movimento, forçando o segundo a movimentar-se simultaneamente. Ao soltar do botão a situação se repete, mas no sentido contrário.

5.3 Discussão dos resultados dos testes

Nesta seção apresentam-se todos os circuitos propostos anteriormente, modelados e simulados utilizando o *software* desenvolvido, e analisa-se os resultados de cada teste. Não se repetem as descrições do comportamento esperado para cada circuito, apenas discorre-se acerca dos resultados pertinentes.

A Figura 5.10 apresenta a sequência simplificada dos estados da simulação do primeiro circuito de teste, que apresenta o comportamento esperado descrito na Seção 5.2.

Quanto ao segundo circuito de testes, a Figura 5.11 exhibe sua sequência de estados, e novamente o sistema apresentou o funcionamento esperado. Além disso, não apresenta o problema da oscilação de ativação do contator *K1* que ocorre no *Automation Studio*.

O terceiro circuito de teste, cujas etapas de simulação são apresentadas na Figura 5.12, também apresentou o funcionamento esperado, descrito na Seção 5.2.

Diferentemente dos outros circuitos designados para testes, o quarto sistema, contido na Figura 5.13 é demasiadamente complexo, e a apresentação da sequência completa de seus estados em simulação estenderia demasiadamente o tamanho deste trabalho. Portanto, optou-se por apresentar somente três estados: o inicial, contido na Figura 5.13a, um estado intermediário arbitrariamente selecionado, revelado na Figura 5.13b, e o resultante do pressionamento do botão de emergência, exibido na Figura 5.13c. Nesse último, é possível verificar o corte na alimentação da parte inferior do diagrama elétrico, e o acionamento do solenoide *Sol3*, que garante o retorno do sistema à sua condição inicial. No mais, os circuitos novamente apresentaram o comportamento esperado.

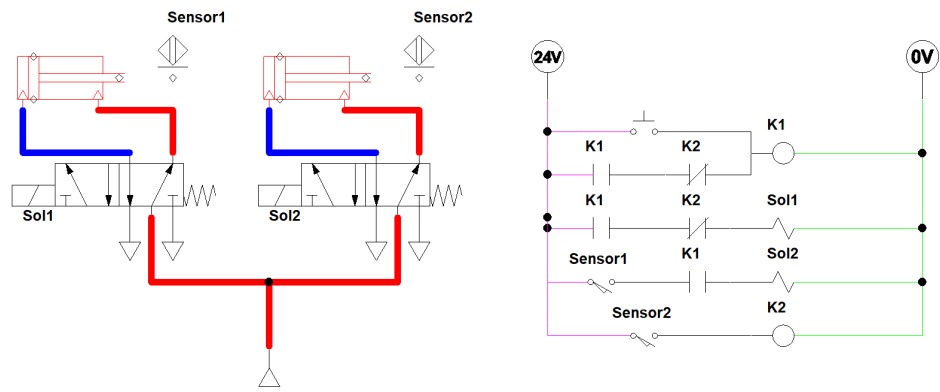
O quinto sistema de teste objetiva a verificação do funcionamento dos cilindros de simples ação, e o resultado de sua simulação está contido na Figura 5.14. Após o fim do pressionamento prolongado do botão, apenas o cilindro com retorno por mola volta à seu estado inicial, conforme esperado.

Ao contrário dos cinco outros circuitos de teste já discutidos, o sexto sistema, cuja sequência de estados é apresentada na Figura 5.15, configura uma limitação do algoritmo de simulação de sistemas fluidos especificado na Seção 4.3. Ao apertar o botão que ativa o solenoide *Sol1*, os dois atuadores deveriam se deslocar simultaneamente, um comportamento verificado no *Automation*

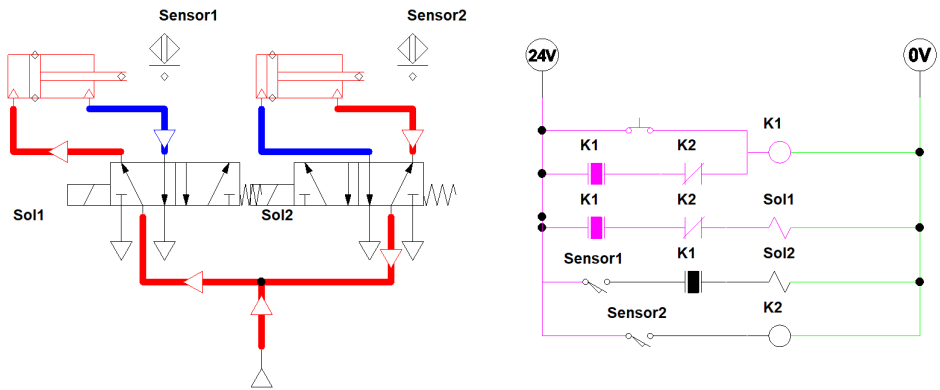
Studio. Entretanto, a Figura 5.15b revela que nenhum dos cilindros se movimenta.

O problema constatado ocorre pois os cilindros de dupla ação movimentam-se apenas ao receber o sinal positivo em qualquer um de seus terminais, e o sinal negativo no outro. Entretanto, o circuito da Figura 5.15 apresenta uma ligação em série entre dois atuadores. Em outras palavras, ambos os cilindros possuem terminais que nunca são conectados a um reservatório, que é a fonte de sinal negativo no algoritmo criado. Logo, não são capazes de movimentar-se.

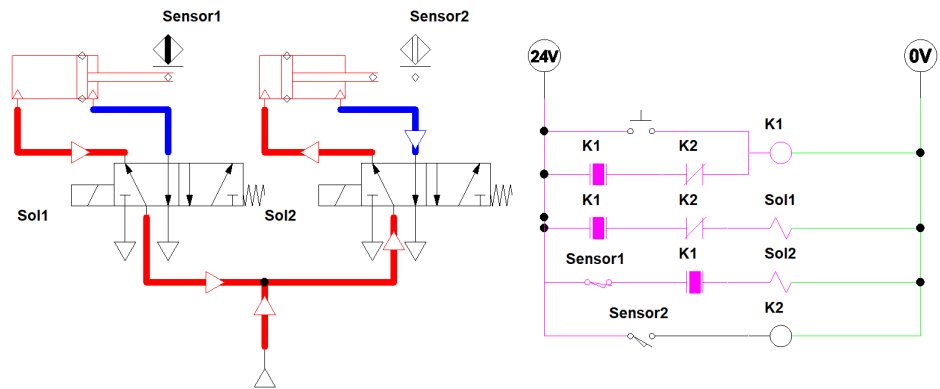
No que diz respeito ao módulo de edição de circuitos desenvolvido, ressalta-se que todos os sistemas propostos para validação do simulador foram modelados com sucesso utilizando-se somente as funções de edição do *software* desenvolvido, ainda que a simulação tenha apresentado comportamentos físicos errôneos no último teste.



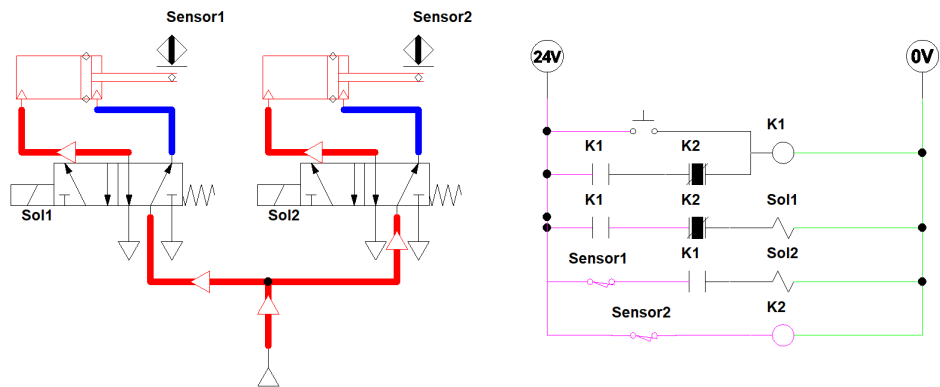
(a) Estado inicial



(b) Estado após pressionamento do botão



(c) Estado após ativação do sensor 1



(d) Estado após ativação do sensor 2

Figura 5.3: Simulação do primeiro circuito de teste

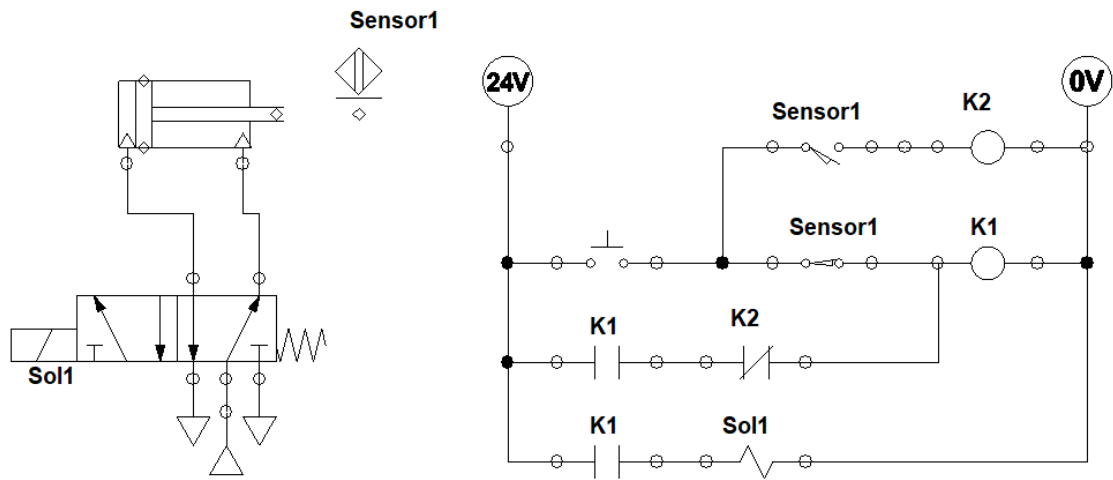


Figura 5.4: Diagramas pneumático e elétrico do segundo circuito de teste

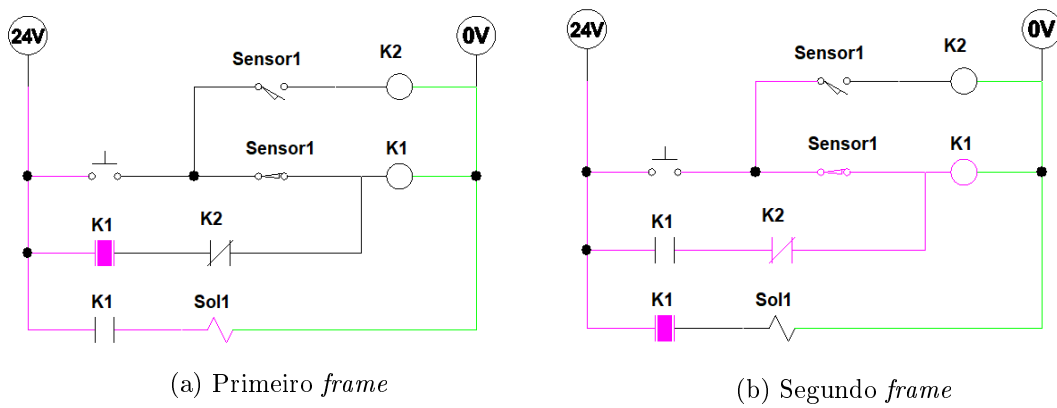


Figura 5.5: Oscilação do contator $K1$ na simulação do teste da Figura 5.4 no *Automation Studio*

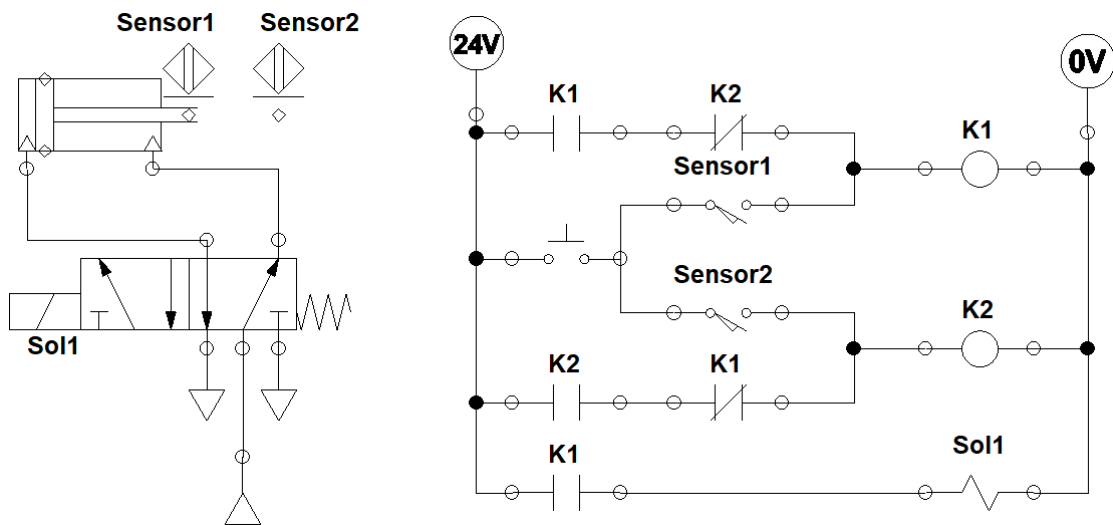


Figura 5.6: Diagramas pneumático e elétrico do terceiro circuito de teste

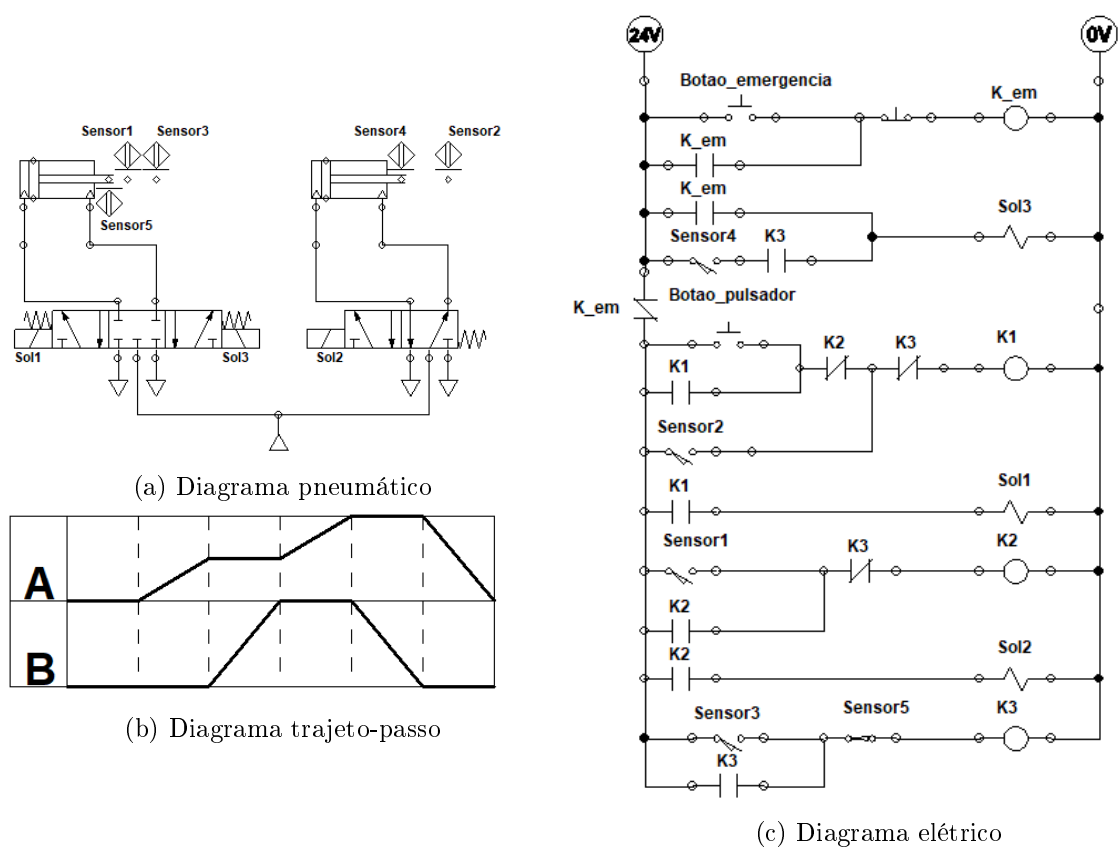


Figura 5.7: Diagramas do quarto sistema de teste

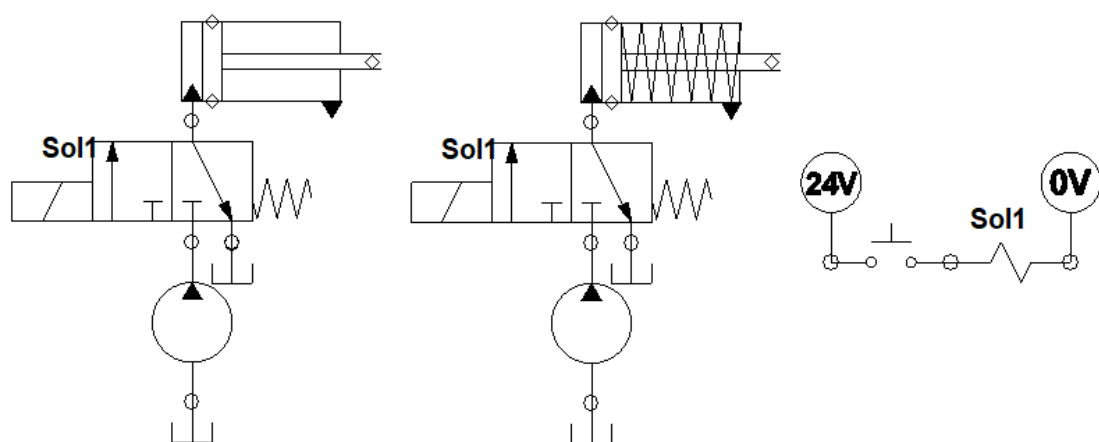


Figura 5.8: Diagramas hidráulico e elétrico do quinto sistema de teste

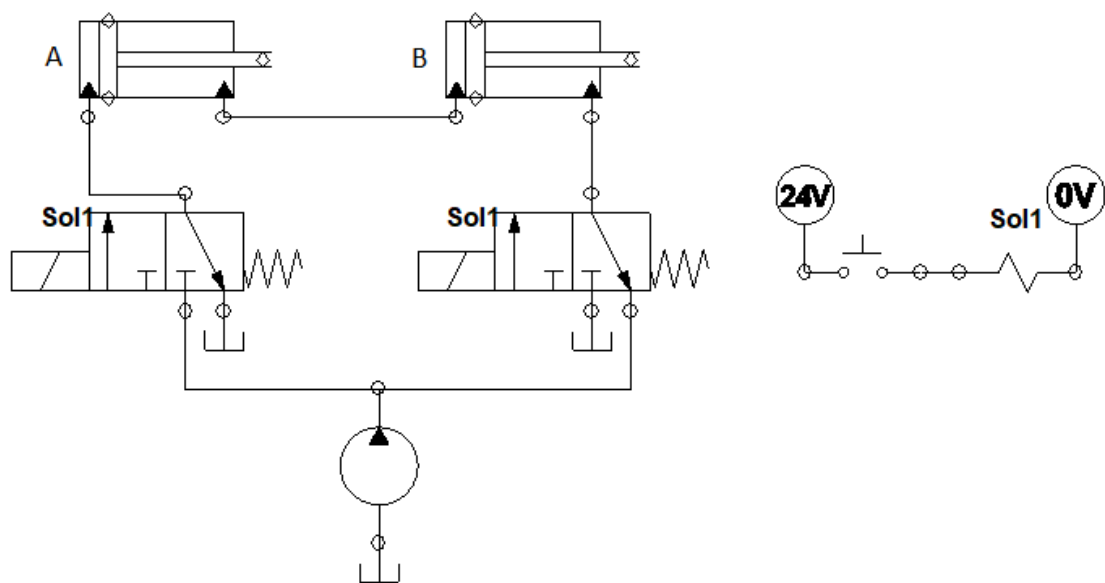
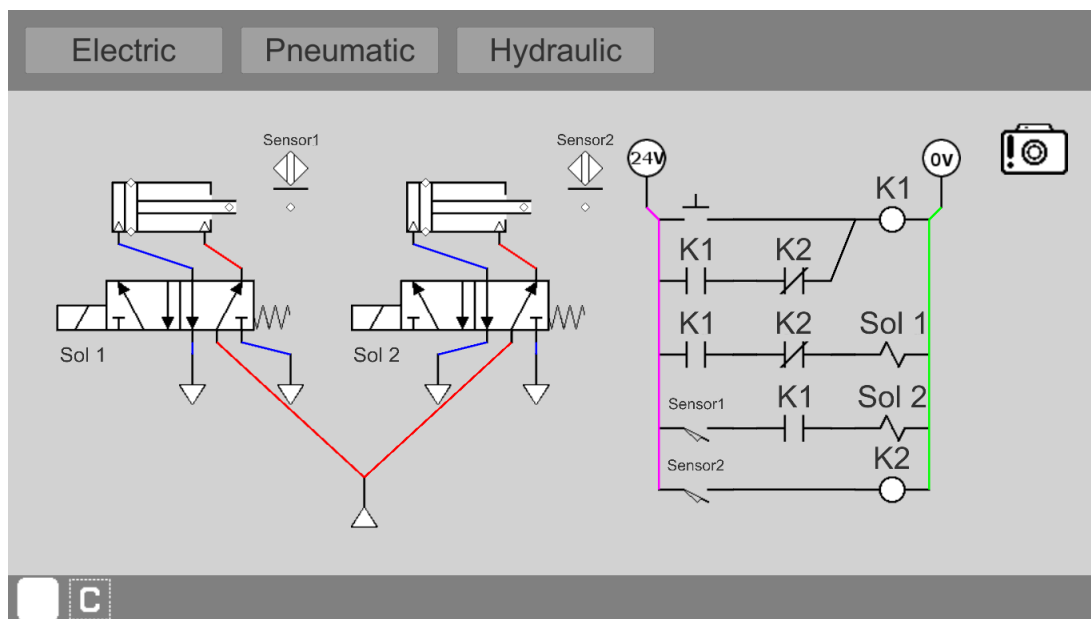
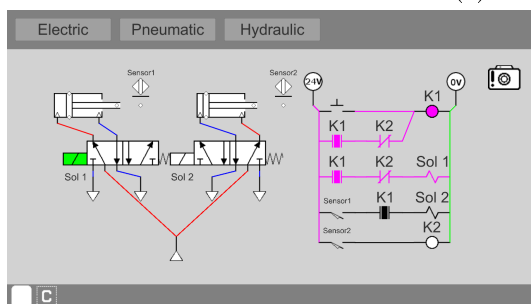


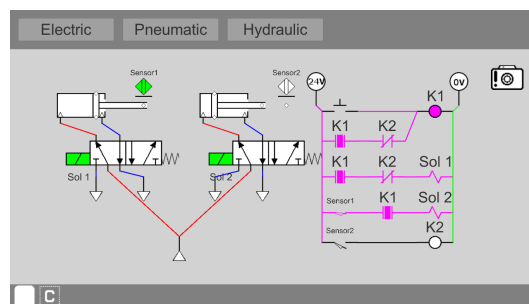
Figura 5.9: Diagramas hidráulico e elétrico do sexto sistema de teste



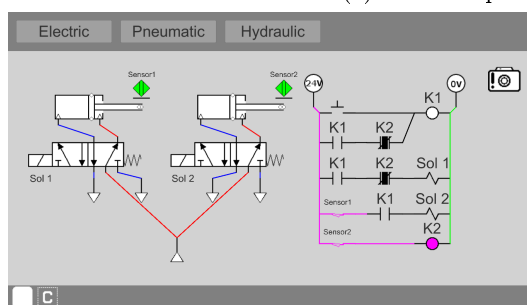
(a) Estado inicial



(b) Estado após pressionamento do botão

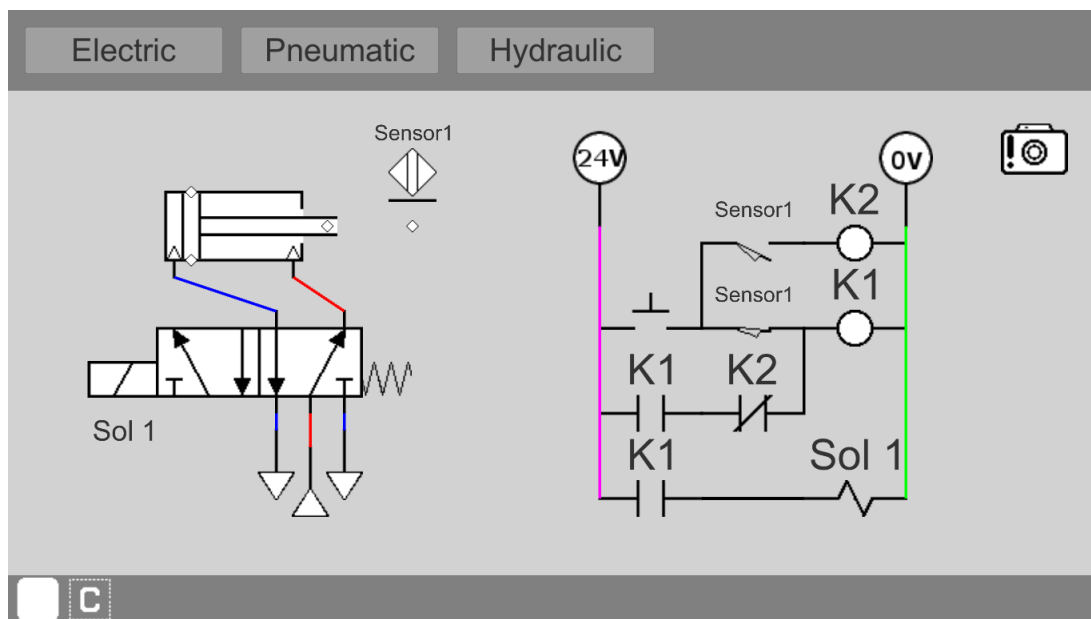


(c) Estado após ativação do sensor 1

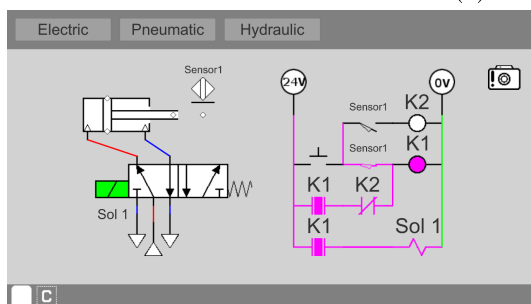


(d) Estado após ativação do sensor 2

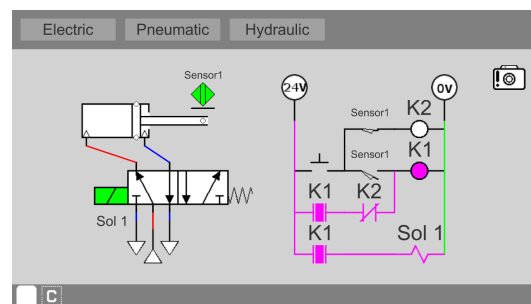
Figura 5.10: Simulação do primeiro circuito de teste



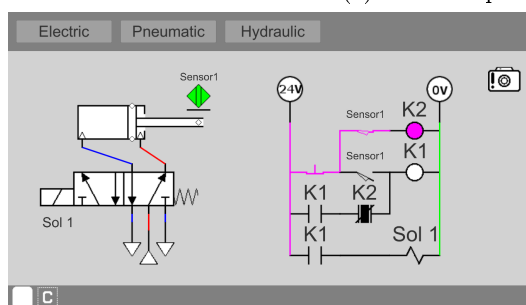
(a) Estado inicial



(b) Estado após pressionamento do botão

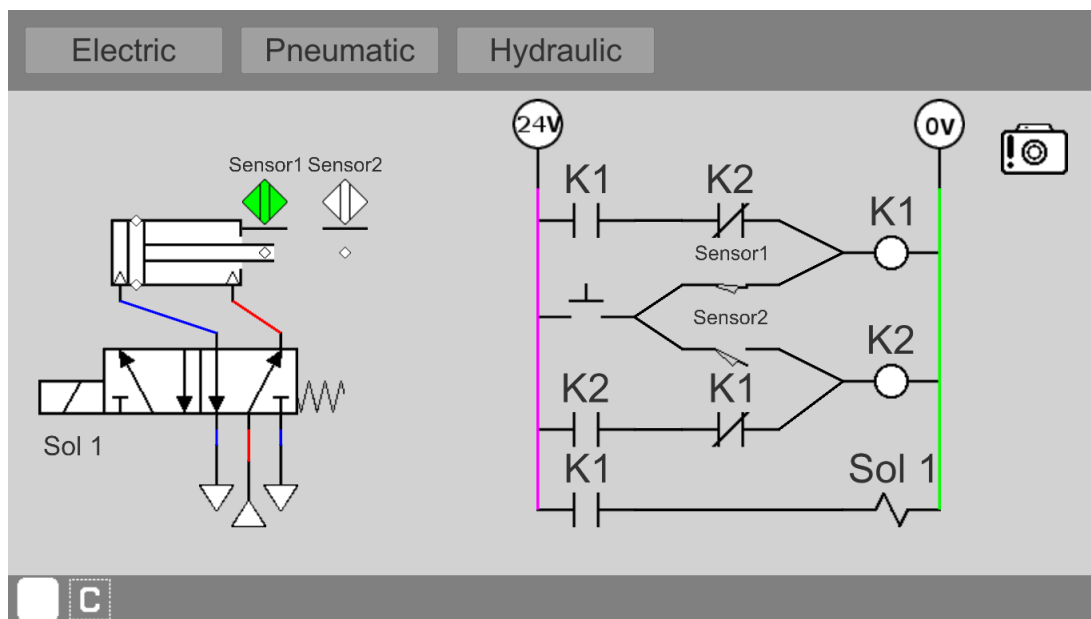


(c) Estado após ativação do sensor 1

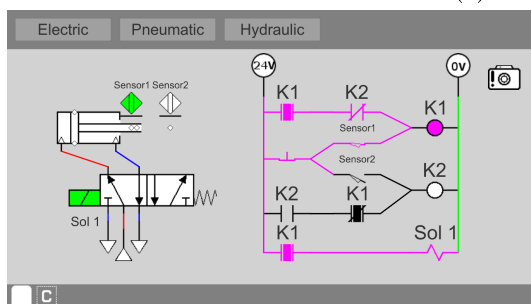


(d) Estado após novo pressionamento do botão

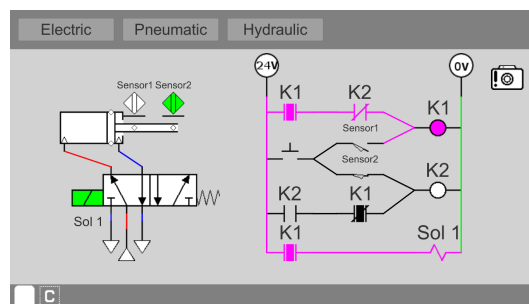
Figura 5.11: Simulação do segundo circuito de teste



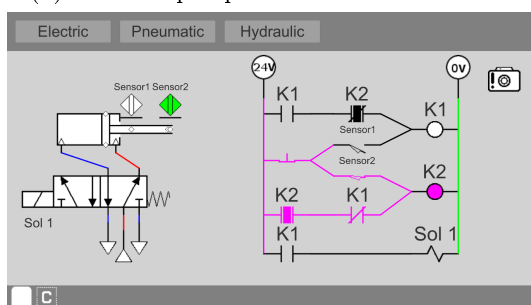
(a) Estado inicial



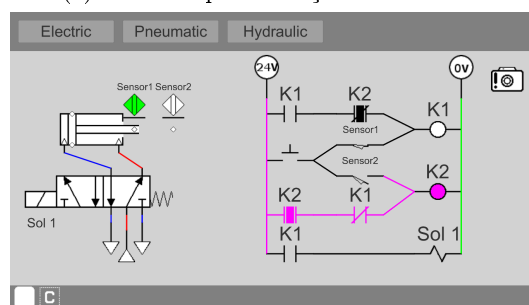
(b) Estado após pressionamento do botão



(c) Estado após ativação do sensor 2

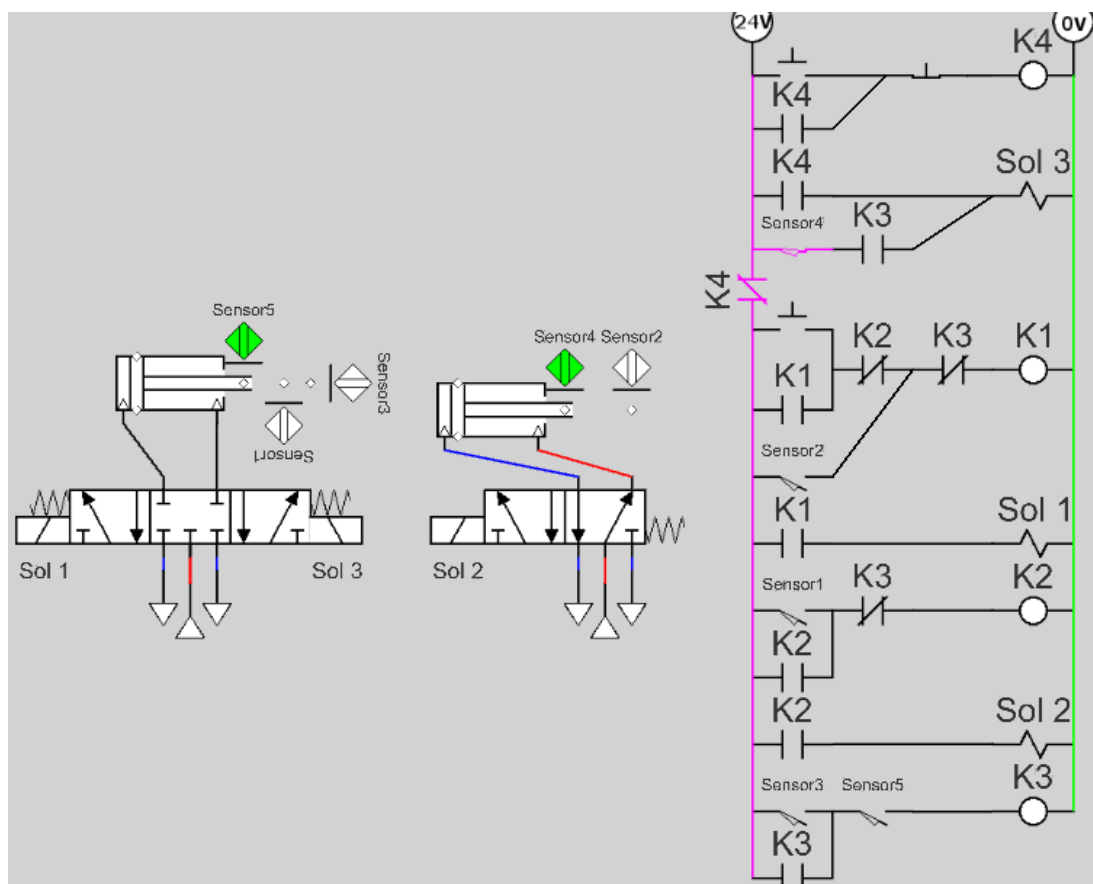


(d) Estado após novo pressionamento do botão

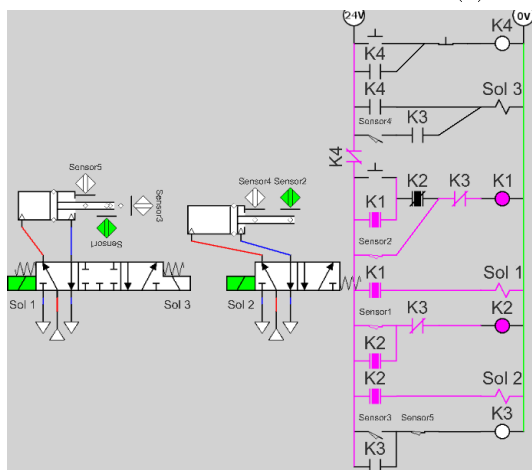


(e) Estado após retorno do cilindro

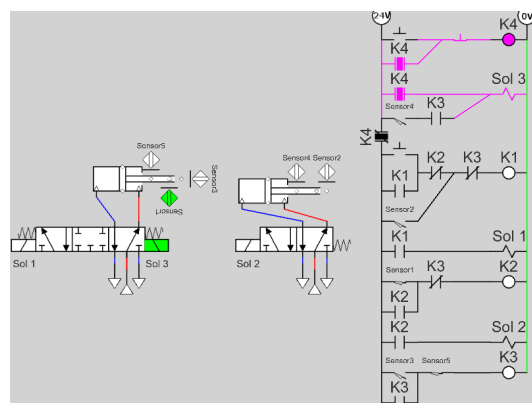
Figura 5.12: Simulação do terceiro circuito de teste



(a) Estado inicial

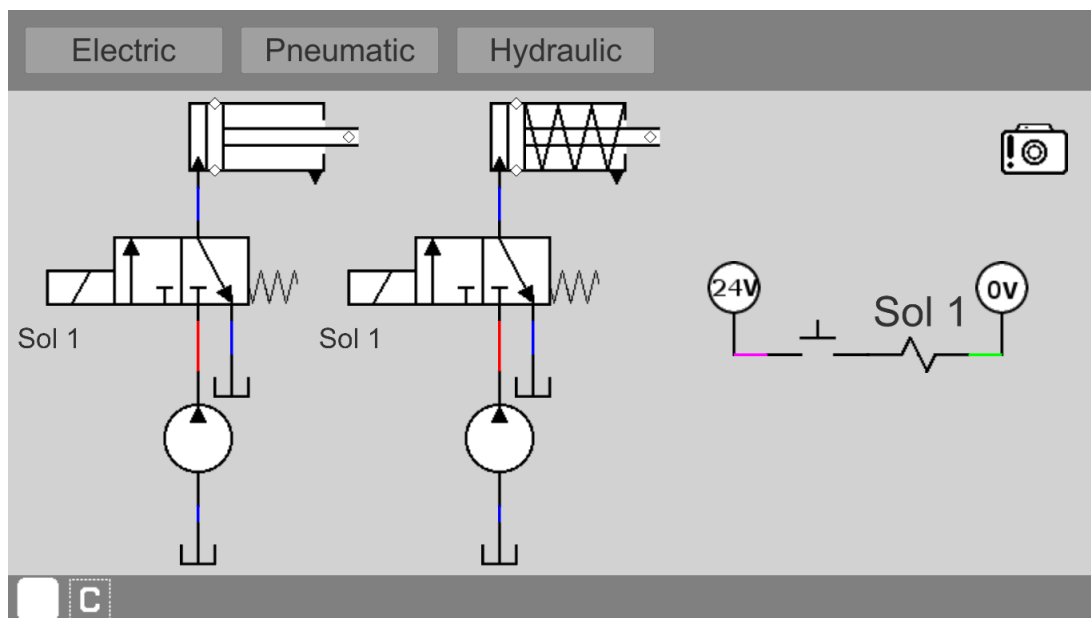


(b) Estado intermediário arbitrário

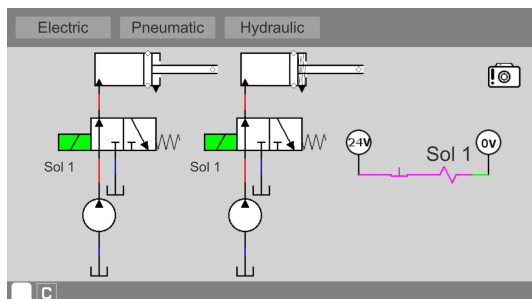


(c) Estado após pressionamento do botão de emergência

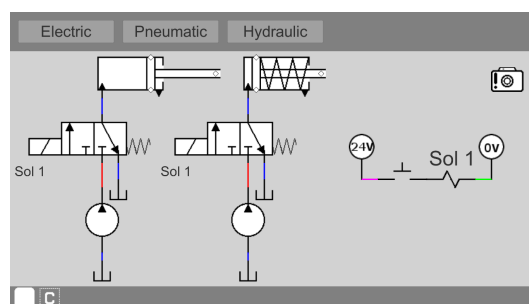
Figura 5.13: Simulação do quarto circuito de teste



(a) Estado inicial

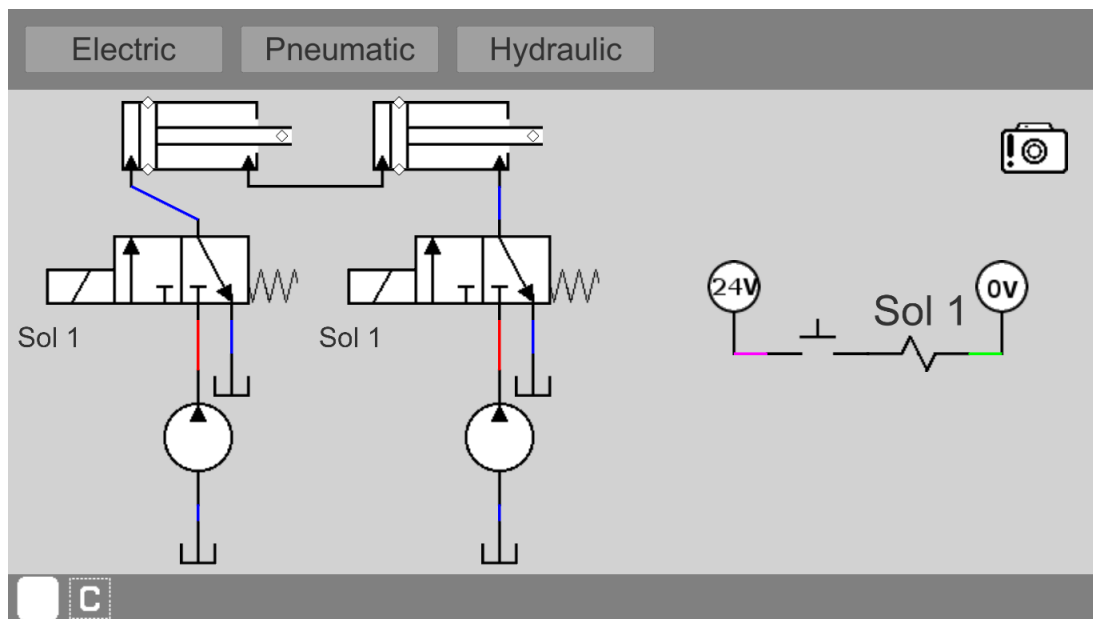


(b) Estado após pressionamento contínuo do botão

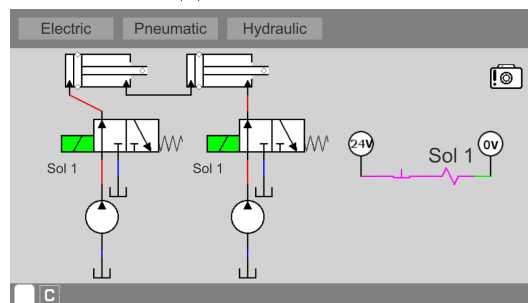


(c) Estado final

Figura 5.14: Simulação do quinto circuito de teste



(a) Estado inicial



(b) Estado após pressionamento do botão

Figura 5.15: Simulação do sexto circuito de teste

Capítulo 6

Conclusões e propostas de trabalhos futuros

Este trabalho apresentou o projeto e implementação de um *software* livre e multiplataforma de edição e simulação de sistemas hidráulicos e pneumáticos, motivado pela necessidade de uma solução acessível de simulação desses circuitos como ferramenta de aprendizado. Como forma de delimitação de escopo, optou-se por implementar o mínimo possível de elementos fluidos e elétricos, e sua modelagem priorizou o entendimento do funcionamento lógico dos circuitos em detrimento de precisão física.

O simulador foi desenvolvido utilizando a *game engine Unity*, e seu projeto foi dividido em dois módulos principais, um de edição e outro de simulação. Um conjunto de circuitos eletropneumáticos e eletro-hidráulicos foi especificado como forma de validação tanto das funcionalidades de implementação de circuitos do módulo de edição quanto dos algoritmos de simulação desenvolvidos, utilizando o *Automation Studio* como referencial de funcionamento.

Todos os circuitos de teste escolhidos puderam ser reproduzidos no simulador. Quanto ao funcionamento do módulo de simulação, dos seis sistemas de teste escolhidos, cinco apresentaram comportamento equivalente ao *Automation Studio*. O sexto circuito de teste não obteve sucesso, consequência das escolhas realizadas na especificação dos algoritmos de simulação dos sistemas fluidos.

A criação de uma interface gráfica simples e intuitiva consiste de um objetivo específico deste trabalho. Embora o correto funcionamento da interface tenha sido testado e validado, não foi realizada uma pesquisa de usabilidade do simulador desenvolvido com alunos de Engenharia e áreas relacionadas, impossibilitando quaisquer declarações acerca desse parâmetro da interface projetada.

Possibilitado pela escolha da *engine Unity* para o desenvolvimento, a mesma versão do simulador foi publicada [42] para cada uma das plataformas contempladas. Dessa forma, o trabalho obteve sucesso no desenvolvimento de uma ferramenta multiplataforma de simulação.

6.1 Trabalhos futuros

Consideradas as limitações apresentadas pelo módulo de simulação desenvolvido, o projeto e implementação de um modelo mais preciso de simulação de sistemas fluido mecânicos é um passo necessário para o aprimoramento da solução de simulação criada neste trabalho.

Ainda mais, a prevalência de soluções proprietárias de simulação na área de hidráulica e pneumática apresenta a oportunidade de especificar padrões generalizados e abertos para a descrição de sistemas fluido mecânicos, além de bibliotecas de elementos hidráulicos, pneumáticos e elétricos.

Outra maneira de aprimorar o simulador desenvolvido neste trabalho consiste na adição de funcionalidades adicionais, como a identificação de conflitos de sensores, expansão da biblioteca de elementos, e estimação do orçamento dos sistemas hidráulicos e pneumáticos simulados. No caso da versão *mobile* do simulador, é interessante adicionar a possibilidade de compartilhar os arquivos de simulação salvos.

Pensando no âmbito educacional, pode-se criar funcionalidades voltadas para o ensino, tais como um modo de “avaliação”, onde os alunos teriam um tempo limite para a implementação de sistemas fluido mecânicos a partir da especificação do funcionamento desejado.

Finalmente, pode-se avaliar a usabilidade do simulador por meio de pesquisa com alunos de Engenharia, de modo a validar a utilidade do simulador como ferramenta de aprendizado de sistemas hidráulicos e pneumáticos.

REFERÊNCIAS BIBLIOGRÁFICAS

- [1] PARR, A. *Hydraulics and Pneumatics: A Technician's and Engineer's Guide*. [S.l.]: Elsevier Ltd., 2011.
- [2] HACKWORTH, J. R. *Programmable Logic Controllers: Programming Methods and Applications*. [S.l.: s.n.].
- [3] SHEPHERD, D. G. *Historical Development of the Windmill*. Último acesso em: 14-06-2018. Disponível em: <<http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.656.3199&rep=rep1&type=pdf>>.
- [4] REYNOLDS, T. S. *Stronger than a Hundred Men: A History of the Vertical Water Wheel*. [S.l.]: Johns Hopkins University Press, 2002.
- [5] YANNOPOULOS, S. I. et al. Evolution of water lifting devices (pumps) over the centuries worldwide. *Water*, set. 2015.
- [6] HYDRAULICS ONLINE. *Hydraulic Press*. Último acesso em: 15-06-2018. Disponível em: <<http://www.hydraulicsonline.com/history-of-hydraulics>>.
- [7] WESTINGHOUSE NUCLEAR. *History of George Westinghouse*. Último acesso em: 15-06-2018. Disponível em: <<http://www.westinghousenuclear.com/About/History>>.
- [8] CHIANG, M.-H.; LEE, L.-W.; HUANG, K.-S. Development of a hydraulic-piezoelectric-actuator for hybrid positioning control with large stroke, high loading and sub-micrometer accuracy. In: *IEEE International Conference on Mechatronics, 2005. ICM '05*. [S.l.: s.n.], 2005. p. 45–49.
- [9] WATTON, J. *Fundamentos de Controle em Sistemas Fluidomecânicos*. [S.l.]: LTC, 2012.
- [10] SILVA, E. C. N. *Apostila de Pneumática*. Último acesso em: 25-06-2018. Disponível em: <<http://sites.poli.usp.br/d/pmr2481/pneumat2481.pdf>>.
- [11] UNIVERSIDADE DE BRASÍLIA. *Ementa da disciplina de Sistemas Hidráulicos e Pneumáticos da UnB*. Último acesso em: 15-06-2018. Disponível em: <<https://matriculaweb.unb.br/graduacao/disciplina.aspx?cod=168238>>.
- [12] UNIVERSIDADE DE BRASÍLIA. *Currículo do curso de Engenharia Mecatrônica da UnB*. Último acesso em: 25-06-2018. Disponível em: <<https://matriculaweb.unb.br/graduacao/curriculo.aspx?cod=6912>>.

- [13] UNIVERSIDADE DE BRASÍLIA. *Currículo do curso de Engenharia Mecânica da UnB*. Último acesso em: 25-06-2018. Disponível em: <<https://matriculaweb.unb.br/graduacao/curriculo.aspx?cod=6424>>.
- [14] FAMIC TECHNOLOGIES. *Automation Studio Educational Home Page*. Último acesso em: 13-06-2018. Disponível em: <<https://www.famicttech.com/edu/index.html>>.
- [15] MATHWORKS. *Simscape Fluids library*. Último acesso em: 13-06-2018. Disponível em: <<https://www.mathworks.com/products/simhydraulics.html>>.
- [16] MSC SOFTWARE. *Easy5: Advanced Controls & Systems Simulation*. Último acesso em: 13-06-2018. Disponível em: <<http://www.mscsoftware.com.br/product/easy5>>.
- [17] SIEMENS. *Simcenter Amesim*. Último acesso em: 13-06-2018. Disponível em: <<https://www.plm.automation.siemens.com/global/en/products/simcenter/simcenter-amesim.html>>.
- [18] FAMIC TECHNOLOGIES. *Automation Studio Home Page*. Último acesso em: 10-09-2018. Disponível em: <<http://www.automationstudio.com/>>.
- [19] FESTO. *FluidSIM 5*. Último acesso em: 10-09-2018. Disponível em: <<https://www.festo-didactic.com/int-en/learning-systems/software-e-learning/fluidsim/fluidsim-5.htm>>.
- [20] BOSCH REXROTH. *Scheme Editor*. Último acesso em: 10-09-2018. Disponível em: <<https://www.boschrexroth.com/en/xc/products/engineering/econfigurators-and-tools/dc-scheme-editor/dc-scheme-editor>>.
- [21] SMC CORPORATION. *Model Selection Software*. Último acesso em: 10-09-2018. Disponível em: <https://www.smc.eu/portal_ssl/webpages/01_products/engineering_tools/model_selection_software/model_selection_software.jsp>.
- [22] SUN HYDRAULICS LLC. *QuickDesign*. Último acesso em: 10-09-2018. Disponível em: <<https://www.sunhydraulics.com/about/highlights/quickdesign-smartconnect-offers-drag-and-drop-schematic-tool>>.
- [23] WOMACK MACHINE SUPPLY COMPANY. *JIC Standard Symbols for Electrical Ladder Diagrams*. Último acesso em: 01-07-2018. Disponível em: <<http://www.womackmachine.com/engineering-toolbox/data-sheets/jic-standard-symbols-for-electrical-ladder-diagrams/>>.
- [24] MATHWORKS. *Pneumatic Pressure Source*. Último acesso em: 01-07-2018. Disponível em: <<https://www.mathworks.com/help/physmod/simscape/ref/pneumaticpressuresource.html>>.
- [25] IMPULSE AUTOMATION LTD. *Pneumatic Proximity Switch*. Último acesso em: 01-07-2018. Disponível em: <<http://www.thepneumaticscompany.com/products/pneumatic-proximity-switch>>.

- [26] FESTO. *Optical Sensors - Festo United States*. Último acesso em: 01-07-2018. Disponível em: <https://www.festo.com/cat/en-us_us/products_050502>.
- [27] FESTO. *Chave Fim de Curso - Festo Didactic*. Último acesso em: 01-07-2018. Disponível em: <<http://www.festo-didactic.com/br-pt/sistemas-de-ensino/bancadas-de-treinamento/pneumatica/eletropneumatica/chave-fim-de-curso.htm?fbid=YnIuchQuNTM3LjIzLjE4LjEwMjAuNTMzMg>>.
- [28] UNITY TECHNOLOGIES. *Game engines - how do they work?* Último acesso em: 25-06-2018. Disponível em: <<https://unity3d.com/what-is-a-game-engine>>.
- [29] UNITY TECHNOLOGIES. *Fast facts - Unity*. Último acesso em 01-07-2018. Disponível em: <<https://unity3d.com/public-relations>>.
- [30] UNITY TECHNOLOGIES. *Activation - Personal: Unity licensing*. Último acesso em 01-07-2018. Disponível em: <<https://unity3d.com/unity/activation/personal>>.
- [31] UNITY TECHNOLOGIES. *Unity - Multiplatform - Publish your game to over 25 platforms*. Último acesso em: 25-06-2018. Disponível em: <<https://unity3d.com/unity/features/multiplatform>>.
- [32] UNITY TECHNOLOGIES. *Unity - Manual: Scenes*. Último acesso em 01-07-2018. Disponível em: <<https://docs.unity3d.com/Manual/CreatingScenes.html>>.
- [33] UNITY TECHNOLOGIES. *Unity - Manual: The Main Windows*. Último acesso em 01-07-2018. Disponível em: <<https://docs.unity3d.com/Manual/UsingTheEditor.html>>.
- [34] UNITY TECHNOLOGIES. *Unity - Manual: AssetWorkflow*. Último acesso em 01-07-2018. Disponível em: <<https://docs.unity3d.com/Manual/AssetWorkflow.html>>.
- [35] UNITY TECHNOLOGIES. *Unity - Manual: Prefabs*. Último acesso em 01-07-2018. Disponível em: <<https://docs.unity3d.com/Manual/AssetWorkflow.html>>.
- [36] UNITY TECHNOLOGIES. *Unity - Manual: UI*. Último acesso em 01-07-2018. Disponível em: <<https://docs.unity3d.com/Manual/UISystem.html>>.
- [37] UNITY TECHNOLOGIES. *Scripting - Unity*. Último acesso em 01-07-2018. Disponível em: <<https://unity3d.com/learn/tutorials/s/scripting>>.
- [38] UNITY TECHNOLOGIES. *Documentation, Unity scripting languages and you - Unity Blog*. Último acesso em 01-07-2018. Disponível em: <<https://blogs.unity3d.com/2014/09/03/documentation-unity-scripting-languages-and-you/>>.
- [39] UNITY TECHNOLOGIES. *Unity - Scripting API*. Último acesso em 01-07-2018. Disponível em: <<https://docs.unity3d.com/ScriptReference/>>.
- [40] JSON. Último acesso em 01-07-2018. Disponível em: <<https://www.json.org/>>.
- [41] UNITY TECHNOLOGIES. *Unity - Manual: JSON Serialization*. Último acesso em 01-07-2018. Disponível em: <<https://docs.unity3d.com/Manual/JSONSerialization.html>>.

[42] NAVES, G. *Repositório do Open Fluid Simulator*. Último acesso em: 01-07-2018. Disponível em: <<https://github.com/gabrielnaves/open-fluid-simulator>>.